OPTIMASI *MULTIPLE TRAVELLING SALESMAN PROBLEM* (M-TSP) PADA PENENTUAN RUTE OPTIMAL PENJEMPUTAN PENUMPANG *TRAVEL* MENGGUNAKAN ALGORITME GENETIKA

SKRIPSI

Untuk memenuhi sebagian persyaratan memperoleh
gelar Sarjana Komputer

Disusun oleh:
Pande Made Rai Raditya
NIM: 135150201111096

PROGRAM STUDI INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2017

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Travel agen merupakan jasa penyedia wisata yang memiliki peranan penting di industri pariwisata Indonesia. Saat ini *travel* menjadi bisnis yang semakin berkembang dari tahun ke tahun. Antusias penumpang menggunakan jasa *travel* dapat dilihat dengan semakin banyaknya perusahaan-perusahaan *travel* yang mudah dijumpai di setiap kota (Samudra & Muklash, 2013). Kota Malang merupakan daerah yang penduduknya cukup padat dikarenakan jumlah mahasiswa yang banyak, begitu juga dengan mahasiswa yang berasal dari luar kota. Hal inilah yang membuat semakin banyaknya jasa *travel* di kota Malang. *Travel* memiliki sistem layanan antar jemput penumpang sampai ke tempat tujuan sesuai dengan trayek atau jurusan yang dilayani (*Door to Door Service*). Jasa *travel* menawarkan fasilitas yang lebih dibandingkan dengan angkutan umum lainnya baik dari segi kenyamanan, keamanan, jadwal kebrangkatan, dan waktu perjalanan yang lebih effisien karena *travel* langsung mengantar penumpang ke alamat tujuan. Dengan kelebihan yang dimiliki, menjadikan *travel* sebagai alat transportasi pilihan penumpang yang cocok digunakan untuk melakukan perjalanan antarkota dibandingkan menggunakan alat transportasi umum lainnya seperti bus dan taxi.

Untuk menjaga kualitas layanan yang diberikan kepada penumpang, pengelola *travel* sebisa mungkin meminimalkan hambatan-hambatan yang ada, salah satunya adalah bagaimana menentukan rute penjemputan penumpang yang optimal. Setiap perusahaan *travel* di kota Malang memiliki banyak armada atau mobil *travel* yang akan menjemput satu persatu penumpang ke alamat yang diminta. Penentuan rute ini bertujuan untuk meminimalkan jarak tempuh dan waktu tempuh perjalanan. Dalam penjemputan penumpang, penentuan rute bergantung sepenuhnya pada pengetahuan supir terhadap lokasi-lokasi penumpang yang akan dijemput sehingga hal ini dirasa kurang optimal. Penentuan rute yang kurang optimal berdampak pada biaya operasional *travel* dan waktu yang tidak effisien sehingga menyebabkan penumpang menunggu terlalu lama dan tidak tepat waktu sampai di tempat tujuan. (Samudra & Muklash, 2013). Hal tersebut dapat menyebabkan kurangnya kualitas pelayanan yang diterima oleh penumpang. Untuk itu dibutuhkan metode yang dapat menyelesaikan permasalahan tersebut.

Salah satu metode yang digunakan untuk dapat menyelesaikan permasalahan diatas yaitu metode algoritme genetika *Multiple Travelling Salesman Problem* (M-TSP). Algoritme genetika dapat menjadi solusi masalah optimasi yang model matematikanya kompleks atau bahkan sulit dibangun (Mahmudy, 2015). Alamat- alamat yang akan dituju oleh sopir setiap harinya merupakan masalah yang

kompleks, karena penumpang memiliki alamat yang berbeda-beda antara satu dengan yang lainnya. Oleh karena itu dalam pengimplementasiannya masalah ini dapat diselesaikan dengan menggunakan algoritme genetika *Multi Travelling Salesman Problem (M-TSP)*. Permasalahan M-TSP memiliki kesamaan dengan TSP, namun pada permasalahan M-TSP menggunakan lebih dari satu orang salesman, dimana dalam kasus ini penjemputan penumpang menggunakan lebih dari satu mobil *travel*.

Terdapat banyak penelitian sebelumnya yang menyatakan algoritme genetika mampu menyelesaikan masalah yang berhubungan dengan pencarian rute, seperti pencarian rute perjalanan terbaik untuk pendistribusian beras bersubsidi menggunakan *Vehicle Routing Problem with Time Window* (VRPTW) yang dilakukan oleh Putri, Mahmudy & Ratnawati (2015). Penelitian sejenis juga dilakukan oleh Sari dan Mahmudy (2015) pada objek yang berbeda, yaitu *Optimasi Multi Travelling Salesman Problem (M-TSP)* pada pendistribusian air mineral. Dalam penelitian ini menggunakan objek *travel* dan menggunakan parameter berupa jarak dan waktu. Dengan menggunakan parameter jarak dan waktu menjadikan penelitian tugas akhir ini berbeda dengan penelitian sebelumnya yang hanya menggunakan parameter jarak sebagai acuan. Penelitian ini juga menggunakan normalisasi data agar parameter jarak dan waktu memiliki pengaruh yang sama dalam proses perhitungan. Normalisasi data yang digunakan pada penelitian ini yaitu *min-max normalization.*

Berdasarkan paparan di atas, penulis mengajukan penelitian berjudul **"Optimasi *Multiple Travelling Salesman Problem* (M-TSP) Pada Penentuan Rute Optimal Penjemputan Penumpang *Travel* Menggunakan Algoritme Genetika"** Tujuan dari penelitian ini adalah membuat suatu sistem cerdas untuk mendapatkan rute optimal pada penjemputan penumpang *travel*. Sistem ini diharapkan nantinya dapat menjadi solusi bagi perusahaan jasa *travel* dalam mengoptimpalkan proses antar jemput penumpang.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka permasalahan yang diangkat pada penelitian ini adalah:

1. Bagaimana mengimplementasikan algoritme genetika untuk optimasi rute optimal penjemputan penumpang *travel* dengan algoritme *multiple travelling salesman problem* (M-TSP)?
2. Bagaimana menentukan representasi kromosom dan fungsi *fitness* untuk optimasi rute optimal penjemputan penumpang *travel*?
3. Bagaimana pengaruh parameter algoritme genetika untuk optimasi penjemputan penumpang *travel*?

1.3 Batasan Masalah

Dari permasalahan di atas, batasan masalah dalam penyusunan penelitian ini adalah:

1. Kendaraan maksimal menampung 5 orang penumpang.
2. Data yang digunakan merupakan data alamat penumpang dari perusahaan Kurnia Agung Selaras *Travel* yang ada di kota Malang.
3. Data jarak dan waktu didapatkan dari hasil prediksi menggunakan *Google Map.*
4. Tempat keberangkatan masing-masing *sales* dimulai dari pangkalan *travel*.
5. Keadaan jalan diasumsikan lancar dan tidak ada macet.

1.4 Tujuan

Adapun tujuan yang ingin dicapai dari penelitian ini adalah:

1. Mengimplementasikan metode algoritme genetika dalam menyelesaikan masalah pemilihan rute optimal penjemputan penumpang *travel*.
2. Menentukan representasi kromosom dan nilai *fitness* untuk optimasi rute penjemputan penumpang *travel*.
3. Mengetahui pengaruh parameter algoritma genetika dalam menyelesaikan permasalahan penjemputan penumpang *travel*.

1.5 Manfaat

Manfaat penelitian yang diharapkan dari penelitian ini adalah:

1. Bagi pemilik *travel*, untuk memberikan kemudahan dalam menenetukan rute yang paling optimal yang akan dikunjungi oleh beberapa sales (sopir *travel*) dalam penjemputan penumpang *travel*.
2. Bagi peneliti, hasil penelitian ini dapat dijadikan acuan penelitian selanjutnya untuk mengembangkan sistem yang lebih kompleks dan memiliki tingkat akurasi yang lebih baik dari penelitian sebelumnya.

1.6 Sistematika Pembahasan

Untuk mencapai tujuan yang diharapkan, penulisan skripsi ini dilakukan berdasarkan sistematika pembahasan sebagai berikut :

BAB 1 PENDAHULUAN

Bab ini berisi latar belakang, rumusan masalah, tujuan, batasan masalah, manfaat, dan sistematika pembahasan.

BAB 2 TINJAUAN PUSTAKA

Bab ini berisi uraian dan pembahasan mengenai teori yang terkait dengan optimasi, *multiple salesman problem*, dan algoritma genetika. Di dalam bab ini juga

terdapat landasan teori dari berbagai sumber pustaka yang terkait.

BAB 3 METODOLOGI

Bab ini berisi penjelasan mengenai langkah-langkah dalam perancangan sistem optimasi penentuan rute penjemputan penumpang *travel* yang terdiri dari studi literatur, pengumpulan data, pengolahan dan analisa data, perancangan metode, dan spesifikasi kebutuhan sistem.

BAB 4 PERANCANGAN

Bab ini berisi penjelasan mengenai rancang bangun sistem yang akan dibuat berdasarkan metode yang digunakan dan perhitungan metode dalam bentuk manualisasi.

BAB 5 IMPLEMENTASI

Bab ini menjelaskan mengenai bagaimana implementasi algoritme yang digunakan terhadap kasus penelitian Optimasi *Multiple Travelling Salesman Problem* (M-TSP) Pada Penentuan Rute Optimal Penjemputan Penumpang *Travel* Menggunakan Algoritme Genetika.

BAB 6 PENGUJIAN DAN PEMBAHASAN

Bab ini berisi penjelasan mengenai pengujian yang dilakukan terhadap penerapan algoritme beserta pembahasan tentang hasil yang telah diperoleh.

BAB 7 PENUTUP

Bab ini berisi kesimpulan yang telah diperoleh dari hasil penelitian dan saran-saran yang bersifat menunjang.

BAB II

TINJAUAN PUSTAKA

2.1 Kajian Pustaka

Penerapan algoritme genetika dalam penyelesaian masalah yang berhubungan dengan pencarian rute telah dilakukan pada beberapa penelitian sebelumnya. Salah satunya oleh Mahmudy & Sari (2015) yang menerapkan algoritme genetika dalam pendistribusian air mineral menggunakan metode *Multiple Travelling Salesman Problem* (M-TSP). Optimasi ini dibentuk guna mendapatkan rute terbaik dalam pendistribusian air mineral ke pelanggan, sehingga waktu dan jarak tempuh dalam pendistribusian air menjadi lebih efisien. Pada proses algoritme genetika ini menggunakan representasi permutasi dengan panjang kromosom sesuai dengan banyaknya pesanan pelanggan, yang setiap angka pada gennya merepresentasikan nomor pelanggan, metode *crossover* yaitu *one-cut point*, metode mutasi dengan *exchange mutation* dan diseleksi dengan *roulette wheel*. Dari hasil pengujian yang dilakukan diperoleh parameter optimal yaitu ukuran populasi sebesar 100 individu dengan rata-rata *fitness* sebesar 0.770, 80 generasi dengan rata-rata *fitness* sebesar 0.772 dan kombinasi *cr* sebesar 0.9 dan *mr* 0.1 dengan rata-rata *fitness* sebesar 0.773. Hasil akhir berupa kombinasi urutan pelanggan yang harus didatangi oleh masing-masing sales beserta dengan total jarak yang paling minimum.

Penelitian mengenai penentuan rute yang menerapkan algoritme genetika juga dilakukan oleh Wijayaningrum & Mahmudy (2016) yang menuliskan tentang optimasi penjadwalan rute kapal, dimana di dalam pembentukan jadwalnya dioptimasikan menggunakan pendekatan algoritme genetika. Di dalam penelitiannya, terdapat sebuah pelabuhan yang memiliki beberapa dermaga (*port*) di mana masing-masing dermaga hanya bisa dikunjungi oleh satu kapal. Penjadwalan dibuat guna mengatur jadwal kunjung kapal menuju dermaga serta meminimalkan jarak antar kapal yang akan berlabuh. Dari hasil perhitungan dan pengujian, didapatkan parameter optimal yaitu jumlah populasi (*popsize*) sebesar 30, jumlah generasi sebesar 100, dan perbandingan nilai *cr* dan *mr* adalah 0,3 dan 0,7. Hasil akhir yang diperoleh adalah berupa optimasi penjadwalan rute kapal yang mampu mengurangi jarak antar kapal yang akan berlabuh.

Pada penelitian yang dilakukan penulis juga menggunakan metode algoritme genetika untuk mengoptimasi rute penjemputan penumpang *travel* di kota Malang. Penentuan rute yang akan dioptimasi ini guna menunjang proses penjemputan penumpang agar lebih efisien dalam hal biaya dan waktu tempuh perjalan. Data yang digunakan dalam penelitian ini berasal dari data salah satu perusahaan *travel* di kota Malang berupa nama dan alamat penumpang. Parameter yang digunakan adalah jumlah penumpang dan jarak antar penumpang. Dalam pembentukan

kromosomnya menggunakan representasi permutasi dari urutan penumpang yang akan dijemput. Dalam proses algoritme genetika, digunakan metode *crossover one cut point*, metode mutasi *exchange mutation,* dan proses seleksinya menggunakan metode *elitsm.*

Perbandingan penelitian sebelumnya dengan penelitian yang akan dilakukan ditunjukkan pada Tabel 2.1.

Tabel 2.1 Kajian Pustaka

Judul	Objek	Metode	Keluaran
	Masukan dan Parameter	**Proses**	**Hasil Penelitian**
*Using Genetic Algorithms to minimize the distance and balance the routes for the Multiple Traveling Salesman Problem (*Raulcezar M. F. Alves & Carlos R. Lopes, 2015)	jarak dan rute yang dilalui sales	*Algoritme genetika*	Rute terpendek
	Jumlah iterasi, ukuran populasi , *crossover rate* (*cr*) dan *mutation rate* (*mr*)	Inisialisasi parameter, proses reproduksi, evaluasi nilai *fitness* dan seleksi	Hasil penelitian menunjukkan algoritme genetika mampu menghasilkan solusi yang baik dengan mengembangkan agoritma menjadi 2 tipe yaitu *multi-objective GA* dan *mono-objective* GA. Tipe *multi-objective GA* cenderung menghasilkan solusi yang lebih baik untuk semua masalah.
Optimizing Multiple Travelling Salesman Problem Considering The Road Capacity (Ponraj,R. & Amalanathan, G., 2014)	jarak, kapasitas jalan (lebar dan luasnya jalan)	*Algoritme genetika*	Rute tependek
	Jumlah iterasi, jumlah sales, jarak antar kota	menentukan alamat ke kota, menetapkan kota untuk salesman dan menemukan rute terpendek untuk setiap salesman	Algoritme Genetika dapat digunakan untuk pemecahan masalah M-TSP dalam memperhitungkan kapasitas jalan.
Penyelesaian *Multiple Travelling Salesperson*	Jarak, durasi perjalanan, dan rute pendistribusian	Algoritme Genetika	Representasi kromosom, perhitungan jarak dan total jarak, hasil crossover, hasil mutasi,

Problem (M-TSP) Dengan Algoritme Genetika : Studi Kasus Pendistribusian Air Mineral (Sari & Mahmudy, 2015)	air mineral		dan hasil seleksi dengan metode *roulette wheel*
	Banyak generasi, jumlah populasi (*popsize*), *crossover rate (cr)* dan *mutation rate*(*mr*)	Pembentukan populasi awal, proses reproduksi, seleksi dengan metode *roulette wheel* dan evaluasi nilai *fitness.*	Algoritme Genetika dapat digunakan untuk penentuan rute optimal pendistribusian air mineral.
Optimization of Ship's Route Scheduling Using Genetic Algorithm (Wijayaningrum & Mahmudy, 2016)	jarak,waktu dan rute kapal	Algoritme genetika	Jadwal rute kapal terbaru
	Jumlah iterasi, ukuran populasi, *crossover rate*(*cr*) dan *mutation rate* (*mr*)	Inisialisasi parameter, pembentukan populasi awal, proses reproduksi, evaluasi nilai *fitness* dan seleksi	Jadwal baru rute kapal yang akan berlabuh yang dapat mengurangi jarak antar kapal yang akan berlabuh
Optimasi Multiple Travelling Salesman Problem (M-TSP) pada Penentuan Rute Optimal Penjemputan Penumpang *Travel* Menggunakan Algoritme Genetika	jarak, waktu, *cost*, dan rute penjemputan penumpang	Algoritme genetika	Representasi kromosom, perhitungan jarak dan total jarak, hasil crossover, hasil mutasi, dan hasil seleksi dengan metode *elitsm.*
	Banyak generasi, jumlah populasi (*popsize*), *crossover rate (cr)* dan*mutation rate*(*mr*)	Pembentukan populasi awal, proses reproduksi, seleksi dengan metode *elitism* dan evaluasi nilai *fitness.*	Rute optimal penjemputan penumpang *travel*.

2.2 Optimasi

Optimasi merupakan suatu proses untuk mencapai hasil yang ideal atau optimal (Akbar & Wicaksana, 2013). Optimasi dapat diartikan sebagai suatu bentuk mengoptimalkan sesuatu hal yang sudah ada ataupun merancang dan membuat sesuatu secara optimal. Teknik optimasi ini diharapkan dapat menemukan titik

maksimum atau titik minimum yang ingin dicari. Untuk mendapatkan solusi yang maksimum, diperlukan data keluaran berupa nilai *fitness,* dimana nilai *fitness* inilah yang akan dijadikan pedoman untuk mencari solusi yang maksimum (Sari & Mahmudy, 2015). Pencarian titik maksimum pada optimasi sering digunakan pada kasus untuk mencari laba terbesar yang akan diperoleh oleh sebuah perusahaan.

Selain digunakan untuk mencari titik maksimum, optimasi juga sering digunakan untuk mencari titik minimum. Pencarian titik minimum pada optimasi biasanya terdapat pada permasalahan *Travelling Salesperson Problem* (TSP). Optimasi didalam *Travelling Salesperson Problem* berarti usaha untuk memperoleh hasil yang optimal pada pencarian rute yang memilik jarak terpendek pada sejumlah daerah untuk dikunjungi seorang salesman (Mahmudy, 2015). Pada permasalahan tersebut, semakin minimum jarak yang ditempuh, maka semakin maksimum keuntungan yang diperoleh. Sebaliknya jika jarak yang ditempuh semakin maksimum, maka semakin minimum keuntungan yang dicapai.

2.3 Algoritme Genetika

Algoritme genetika merupakan salah satu dari bentuk algoritme evolusi yang banyak digunakan dalam pemecahan permasalahan yang kompleks (Mahmudy, 2013). Di dalam proses optimasinya, algoritme genetika menggunakan sebuah teknik optimasi berbasis populasi yang menerapkan tahapan evolusi biologi. Pendekatan yang diambil oleh algoritme genetika adalah dengan menggabungkan secara acak berbagai pilihan solusi terbaik di dalam suatu populasi untuk mendapatkan generasi solusi terbaik yaitu pada suatu kondisi dengan nilai *fitness* yang paling tinggi. Setiap generasi akan merepresentasikan perbaikan-perbaikan pada populasi awalnya. Proses tersebut dilakukan secara berulang sehingga dapat mensimulasikan proses evolusi yang semakin baik. Pada akhir proses akan didapatkan solusi-solusi yang paling tepat dimana akan direpresentasikan sebagai kromosom. Hasil akhir dari algoritme genetika adalah menampilkan kromosom yang memiliki nilai *fitness* tertinggi dari semua generasi. Parameter algoritme genetika yang digunakan pada penelitian ini adalah:

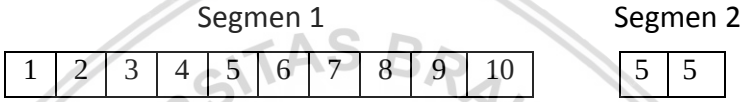
a. *PopSize* yaitu ukuran populasi pada setiap generasi.
b. *Crossover Rate* (*Cr*) adalah kemungkinan terjadinya persilangan (*crossover*) pada suatu generasi.
c. *Mutation Rate* (*Mr*) adalah kemungkinan terjadinya *mutasi* pada setiap individu.
d. Jumlah generasi yang akan dibentuk, jumlah generasi akan menentukan lama penerapan algoritme genetika.

2.4 Proses Algoritme Genetika

Secara umum langkah-langkah untuk menyelesaikan masalah dengan menggunakan algoritme genetika (Mahmudy, 2013) yaitu :

2.4.1 Inisialisasi

Inisialasi merupakan proses yang dilakukan untuk membangitkan himpunan solusi baru secara *random* yang di mana di dalamnya terdapat sejumlah *string* kromosom ke dalam sebuah populasi. Populasi yang dibentuk disesuaikan dengan masalah yang ingin dipecahkan. Pada tahap ini, ukuran populasi (*popsize*) harus ditentukan terlebih dahulu berdasarkan banyaknya individu yang ingin ditampung. Representasi kromosom yang digunakan pada penelitian ini menggunakan representasi kromosom permutasi, yaitu sebuah teknik representasi yang digunakan untuk masalah yang berhubungan dengan pengurutan data (*ordering problem*) atau permasalahan yang berhubungan dengan pengurutan tugas (*task ordering problem*). Representasi nilai pada setiap kromosom berupa barisan angka dalam sebuah urutan. Berikut merupakan contoh representasi kromosom permutasi yang ditunjukkan pada Gambar 2.1



Gambar 2.1 Contoh Representasi Kromosom Permutasi

Representasi kromosom pada Gambar 2.1 memiliki dua segmen. Segmen 1 menunjukan banyaknya lokasi penjemputan penumpang dan segmen 2 menunjukkan jumlah penumpang yang dijemput setiap *salesman*, dimana panjang kromosom pada segmen 1 tergantung dari banyaknya lokasi penjemputan penumpang.

2.4.2 Reproduksi

Setelah terbentuknya populasi, dilanjutkan dengan proses reproduksi yang bertujuan untuk menghasilkan ketuturunan berikutnya (*offspring*). Terdapat dua cara proses reproduksi pada algoritme genetika, yaitu pindah silang (*crossover*) dan mutasi (*mutation*).

2.4.2.1 Crossover

Crossover adalah proses kawin silang antara dua individu yang terpiih secara acak dengan menentukan *cut point* (titik potong) terlebih dahulu. Tujuan dari proses crossover ini adalah untuk memperoleh keturunan yang otomatis akan memperbanyak variasi individu pada populasi tersebut.

Secara umum proses crossover pada algoritme genetika dilakukan dengan cara memotong kromosom parent 1 pada *cut point* tertentu kemudian menggabungkannya dengan potongan kromosom parent 2 sehingga didapatkan *offspring* 1 dan sebaliknya memotong kromosom parent 2 pada *cut point* tertentu kemudian menggabungkannya dengan potongan kromosom parent 1 sehingga didapatkan child *offspring* 2. Sehingga sifat keturunan (*offspring*) tidak akan jauh berbeda dengan kedua induknya karena kromosom anak merupakan kombinasi dari

kromosom induknya. Misalkan terdapat dua individu yang terpilih secara acak. Maka pada crossover ini akan dihasilkan dua *offspring* sebagai berikut:

Cut point

Parent 1	6	8	7	3	1	4	10	5	9	2
Parent 2	4	1	9	2	3	5	6	7	8	10
Offspring 1	6	8	7	3	1	4	9	2	5	10
Offspring 2	4	1	9	2	3	6	8	7	10	5

Gambar 2.1 Contoh *Crossover One Cut Point*

Metode crossover pada Gambar 2.1 merupakan metode *one cut point crossover*. Dari contoh diatas, jika terdapat *crossover rate* = 0.2 dengan *popsize* = 10 maka *offspring* yang diperoleh 0.2 x 10 = 2 *offspring*. Selain metode *one cut point crossover*, metode lain untuk melakukan proses *crossover* adalah *single-point crossover, multi-point crossover, uniform crossover,* dan *permutation crossover* (Nisa, 2013).

2.4.2.2 Mutation

Mutation (mutasi) adalah suatu proses untuk mengubah secara acak salah satu dari bit kromosom. Proses mutasi berfungsi untuk menjadikan individu-individu dalam populasi semakin bervariasi. Karena perubahan yang terjadi adalah perubahan salah satu bit kromosom secara acak, maka individu yang terbentuk kemungkinannya akan memiliki perbedaan yang signifikan dengan sifat induknya. Misalkan terdapat individu yang terpilih yaitu P1. Titik yang terpilih secara acak adalah titik ke 3 dan ke 7. Maka dihasilkan *offspring* baru (C1) sebagai berikut:

P1	1	10	8	4	2	9	5	6	3	7
Offspring C1	1	10	5	4	2	9	8	6	3	7

Gambar 2.2 Contoh *Reciprocal Exchange Mutation*

Metode crossover pada Gambar 2.2 merupakan metode *reciprocal exchange mutation.* Jika dalam proses mutasi sebelumnya telah ditentukan *mutation rate* = 0.1 dengan *popsize* = 10 maka dihasilkan 0.1 x 10 = 1 *offspring.* Sehingga dari contoh diatas akan dilakukan proses mutasi sebanyak satu kali. Metode lain yang digunakan untuk proses mutasi adalah *insertion point*. Metode ini bekerja dengan memilih satu posisi (*selected point* / SP) secara *random* kemudian mengambil dan menyisipkan nilainya pada posisi lain (*insertion point*/ IP) secara *random* (Mahmudy, 2013).

2.4.3 Evaluasi

Evaluasi merupakan proses pengumpulan semua individu dalam populasi beserta dengan *child*nya kemudian dihitung *fitness* dari masing-masing kromosom. Pada proses ini dilakukan pencarian nilai *fitness* dari setiap individu. Di mana nilai *fitness* yang diperoleh akan mewakili kualitas solusi dari individu yang terbentuk (Widodo & Mahmudy, 2010). Semakin besar *fitness* yang dihasilkan, maka semakin baik kromosom tersebut untuk dijadikan calon solusi (Mahmudy, 2013). Setelah proses evaluasi untuk perbaikan populasi, maka generasi-generasi baru ini akan menggantikan himpunan populasi asal. Siklus ini akan berlangsung berulang kali sampai tidak dihasilkan perbaikan keturunan, atau sampai kriteria optimum ditemukan (Mahmudy, 2008).

2.4.4 Seleksi

Seleksi adalah proses pemilihan individu terbaik dari semua alternatif solusi untuk diambil beberapa individu yang memiliki nilai *fitness* yang diharapkan. Tujuan dari proses seleksi adalah untuk memperoleh induk yang terbaik didalam populasi tersebut. Karena jika induk memiliki kromosom yang baik, maka kemungkinan anak yang dihasilkan juga akan memiliki kromosom yang baik pula. Seleksi memiliki berbagai macam metode yaitu *roulette wheel*, *binary tournament*, dan *elitism*. Sebagai contoh *elitism selection* di mana proses seleksi ini mengutamakan perangkingan individu berdasarkan niai *fitness* terbesar ke terkecil yang kemudian akan dijadikan langsung sebagai generasi berikutnya.

2.4.5 *Fitness*

Perhitungan nilai *fitness* merupakan perhitungan seberapa baik individu tersebut jika dijadikan sebuah solusi, dimana nilai *fitness* yang diperoleh akan mewakili kualitas solusi dari individu yang terbentuk (Widodo & Mahmudy, 2010). Nilai *fitness* menyatakan nilai dari kebaikan solusi yang dihasilkan oleh tiap individu dalam satu populasi. Menurut Mahmudy (2015), terdapat dua persamaan untuk melakukan perhitungan *fitness* yaitu *fitness* untuk maksimasi dan untuk minimasi. Berikut merupakan persamaan *fitness* minimasi yang ditunjukan pada persamaan 2.1 dan persamaan 2.2 merupakan persamaan *fitness* untuk maximasi.

$$fitness = \frac{c}{f(x)} \quad \text{..... (2.1)}$$

$$fitness = c * f(x) \quad \text{..... (2.2)}$$

2.5 Normalisasi *Min-Max*

Normalisasi merupakan suatu cara untuk melakukan penyeimbangan antara beberapa parameter dimana parameter-parameter tersebut sama pentingnya dalam suatu metode (Gajera et all, 2016). Pada penelitian ini, data jarak dan waktu yang

didapatkan melalui google maps akan di normalisasi menggunakan proses normalisasi *min-max*, sehingga data jarak dan data waktu yang digunakan sebagai parameter memiliki keseimbangan nilai atau pengaruh yang sama dalam proses perhitungan. Rumus *min-max normalization* (Fatkhiyah, 2012) ditunjukkan sebagai berikut :

$$N' = \frac{N - Min}{Max - Min}(New\_Max - New\ Min) + New\_Min \ldots\ldots\ldots (2.3)$$

Keterangan :
N' : nilai pemetaan
N : nilai original
Min : nilai batas terendah
Max : nilai batas tertinggi
New_Max : nilai batas terbesar dalam pemetaan
New_Min : nilai batas terkecil dalam pemetaan

2.6 *Multiple Travelling Salesman Problem* (M-TSP)

Permasalahan M-TSP merupakan masalah optimasi yang merupakan perluasan dari masalah *Travelling Salesman Problem* (TSP). Permasalahan ini dapat dijadikan acuan dalam mencari rute terpendek maupun melakukan penjadwalan berbagai permasalahan optimasi (Saptaningtyas, 2012). *Multiple Travelling Salesman Problem* (M-TSP) merupakan pengembangan dari *Traveling Salesman Problem* (TSP). Pada permasalahan M-TSP terdapat lebih dari seorang salesman (Singh & Lodhi, 2014).

Multi Travelling Salesman Problem (M-TSP) adalah suatu metode yang digunakan untuk mencari rute jarak terpendek yang harus dilewati oleh beberapa orang sales dalam mengunjungi beberapa tempat tujuan dari suatu tempat asal. Representasi permutasi yang digunakan untuk TSP bisa dimodifikasi sehingga bisa digunakan untuk M-TSP. *Multi Travelling Salesman Problem* (M-TSP) memiliki konsep yang hampir sama dengan TSP, akan tetapi bedanya adalah terletak pada sales yang melakukan kunjungan lebih dari satu orang. Sehingga diperlukan konsep yang lebih dalam lagi untuk dapat menyelesaikan permasalahan M-TSP. Jika jumlah sales dan tempat kunjungan semakin banyak, maka alternatif solusi yang terbentuk akan semakin banyak pula. Permasalahan M-TSP dapat digambarkan sebagai berikut :

posisi	1	2	3	4	5	6	7	8	9	10
gen	2	5	1	7	6	10	4	9	8	3

segmen 1

11	12	13
3	4	3

segmen 2

Gambar 2.5 Contoh chromosome untuk M-TSP

Gen-gen yang ada pada segmen 1 (posisi 1 sampai 10) menunjukkan urutan daerah yang dikunjungi, segmen 2 (posisi 11 sampai 13) menunjukkan banyaknya daerah yang dikunjungi tiap salesman. Dengan asumsi bahwa tiap salesman berangkat dari kantor pusat (KP) dan harus kembali lagi ke kantor pusat, maka dari Gambar 2.5 dihasilkan rute tiap salesman sebagai berikut:

Salesman 1 : KP → daerah 2 → daerah 5 → daerah 1 → KP
Salesman 2 : KP → daerah 7 → daerah 6 → daerah 10 → daerah 4 → KP
Salesman 3 : KP → daerah 9 → daerah 8 → daerah 3 → KP

M-TSP dapat diselesaikan dengan metode heuristik salah satunya dengan algoritme genetika. Kebanyakan metode heuristik yang digunakan yaitu menerima solusi baru yang lebih baik, namun pada metode algoritme genetika solusi baru yang lebih buruk kadang-kadang dapat diterima, sehingga solusi dapat terhindar dari nilai maksimum lokal. Tujuan yang ingin dicapai adalah mendapatkan rute dengan jarak optimal pada penjemputan penumpang *travel*. Hasil penelitian ini diharapkan dapat memberikan alternatif penyelesaian M-TSP sehingga dapat dijadikan pertimbangan untuk bidang usaha yang menerapkan M-TSP dalam aktivitas kerjanya, salah satunya jasa perusahaan *travel* di kota Malang.

2.7 *Travel*

Berkembangan bidang pariwisata di Indonesia menjadikan bisnis jasa transportasi wisata atau *travel* menjadi bisnis yang cukup diminati untuk dikembangkan. Bisnis jasa *travel* merupakan salah satu jenis kegiatan usaha yang bergerak di bidang periwisata yang cukup dikenal di kalangan masyarakat umum. Sebagai penyedia jasa transportasi, *travel* menawarkan berbagai fasilitas untuk mengatur perjalanan wisata ke berbagai daerah. Jasa *travel* menawarkan fasilitas yang lebih dibandingkan dengan angkutan umum lainnya baik dari segi kenyamanan, keamanan, jadwal kebrangkatan, dan waktu perjalanan yang lebih effisien karena *travel* langsung mengantar penumpang ke alamat tujuan. Dengan kelebihan yang dimiliki ini menjadikan *travel* sebagai alat transportasi pilihan penumpang yang hendak melakukan perjalanan antar kota dibandingkan dengan jenis alat transportasi lainnya (Samudra & Muklash, 2013).

Travel memiliki sistem layanan antar jemput penumpang sampai ke tempat tujuan sesuai dengan trayek atau jurusan yang dilayani (*Door to Door Service*). Penjemputan penumpang *travel* dilakukan oleh sopir ke alamat penumpang sesuai dengan waktu keberangkatan. Sopir akan menjemput satu persatu penumpang dan mengantarkannya ke tempat tujuan. Alamat- alamat yang akan dituju oleh sopir setiap harinya merupakan masalah yang kompleks, karena penumpang memiliki

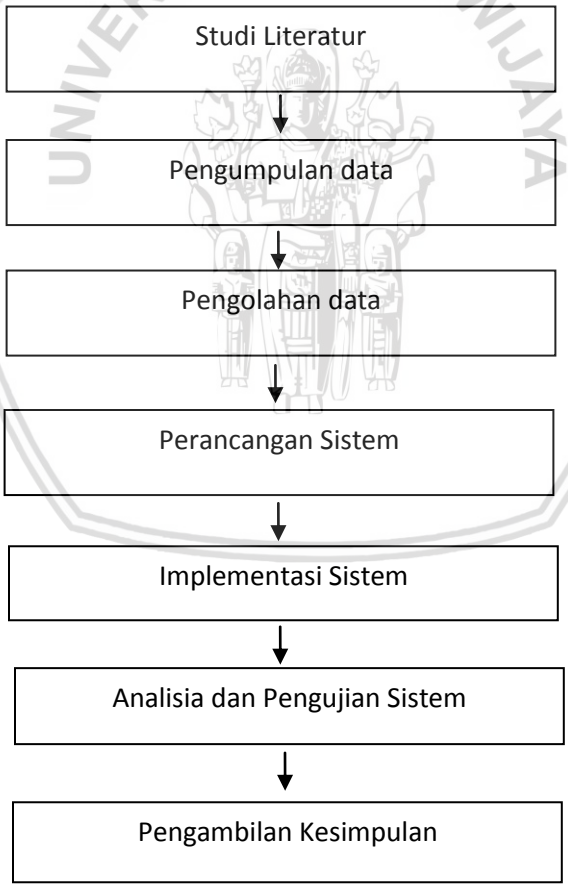
alamat yang berbeda-beda antara satu dengan yang lainnya. Oleh karena itu dalam pengimplementasiannya masalah ini dapat diselesaikan dengan menggunakan algoritme genetika *Multi Travelling Salesman Problem* (M-TSP). Pada permasalahan ini menggunakan lebih dari satu orang *salesman*, dimana dalam kasus ini penjemputan penumpang menggunakan lebih dari satu mobil *travel* dan keberangkatkan dimulai dari pangkalan *travel*. Penentuan rute penjemputan *travel* menjadi salah satu masalah yang sangat penting untuk dipecahkan karena berpengaruh terhadap waktu dan biaya operasional kendaraan. Pada penelitian ini penulis akan menggunakan Optimasi *Multiple Travelling Salesman Problem* (M-TSP) untuk menyelesaikan masalah penentuan rute optimal pada penjemputan penumpang *travel* di wilayah kota Malang.

2.8 Kondisi Jalan

Pada kegiatan transportasi tentunya ada banyak masalah yang dapat terjadi saat melakukan perjalanaan. Salah satu faktor yang menjadi penghambat dalam melakukan perjalanan adalah kondisi jalan. Kondisi jalan yang mengalami kerusakan bisa disebabkan oleh beban lalu lintas yang berlebihan (*overloaded*), panas atau suhu udara, air dan hujan, serta mutu awal produk jalan yang jelek. Kondisi jalan sangat berpengaruh pada kenyamanan pengemudi (*riding quality*). Selain kerusakan jalan, kemacetan lalu lintas juga menjadi faktor yang dapat menyebabkan terhambatnya aktivitas perjalanan. Kemacetan lalu lintas menjadi permasalahan sehari-hari yang sering ditemukan. Kemacetan pada ruas jalan ataupun adanya jalur satu arah pada daerah tertentu akan menghambat kelancaran lalu lintas. Perlu adanya perhatian khusus dari pemerintah untuk mengatasi kemacetan lalu lintas karena hal ini sangat berpengaruh terhadap kegiatan transportasi salah satunya yaitu jasa *travel*. Ada beberapa langkah yang dapat dilakukan untuk memecahkan permasalahan kemacetan lalu lintas antara lain memperlebar luas jalan, menambah lajur lalu lintas, memperbaiki jalan yang mengalami kerusakan, mengubah sirkulasi lalu lintas dan memberikan sanksi jika ada pengemudi yang melanggar. Beberapa hal tersebut dapat dilakukan guna mengurangi kemacetatan lalu lintas.

BAB 3 METODOLOGI

Pada bab ini menjelaskan tentang metodologi penelitian dan perancangan sistem yang akan digunakan pada optimasi penentuan rute optimal penjemputan penumpang *travel* menggunakan algoritme genetika *Multi Travelling Salesman Problem* (M-TSP). Tahap-tahap penelitian dimulai dari mengidentifikasi studi literature, pengumpulan data, pengolahan data, perancangan sistem, implementasi, pengujian dan analisis, dan evaluasi. Tahap dibawah menjelaskan tentang langkah-langkah dan rancangan yang digunakan dalam pembuatan sistem optimasi *multiple travelling salesman problem* (M-TSP) pada penentuan rute optimal penjemputan penumpang *travel* menggunakan algoritme genetika. Tahapan penelitian yang dilakukan pada penelitian ini ditunjukkan pada gambar 3.1.



Gambar 3.1 Diagram Alir Metode Penelitian

3.1 Studi Literatur

Tahap pertama dalam penelitian ini adalah mempelajari literatur dan metode yang digunakan dengan mengumpulkan informasi atau refrensi. Tujuan dari tahapan ini adalah untuk memperoleh informasi mengenai dasar teori dan sumber yang berkaitan guna menunjang penelitian dan pengembangan sistem. Sumber informasi dan literatur yang digunakan bisa didapatkan dari dari buku, jurnal penelitian, situs di internet, penjelasan dari dosen pembimbing, serta referensi lainnya yang bisa digunakan untuk membantu menyelesaikan skripsi ini.

3.2 Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data untuk mendapatkan data sampel sebagai acuan untuk pengembangan sistem. Pada penelitian ini data yang digunakan sebagai objek penelitian adalah data dari salah satu perusahaan *travel* di kota Malang yaitu Kurnia Agung Selaras *Travel*. Data yang diperoleh berupa data nama dan alamat penumpang, dimana penjemputan penumpang di mulai dari kantor *travel*. Untuk menghitung jarak dari satu tempat ke tempat lainnya maka diperlukan peta kota Malang dan sekitarnya menggunakan data jarak dan waktu tempuh yang diambil dari *Google Maps* sehingga data yang diperoleh akan lebih jelas dan akurat. *Google Maps* merupakan sebuah peta digital yang telah dikembangkan oleh *google* yang akan menampilkan denah dari daerah tertentu sesuai dengan skala peta tertentu. Data yang diperoleh kemudian akan dilakukan pengolahan sehingga akan siap untuk diproses dengan penerapan algoritme genetika. Pada penelitian ini menggunakan data set sebanyak 30 alamat penumpang dengan menggunakan 6 unit mobil *travel* yang selanjutnya akan di optimasi menggunakan algoritme genetika dimana hasil optimasi berupa rute terbaik dalam penjemputan penumpang.

3.3 Pengolahan data

Pada tahap ini dilakukan pengolahan data untuk menganalisis data dari studi kasus yang akan di optimasi dengan metode algoritme genetika sehingga menemukan hasil akhir yang mendekati optimum. Dalam pengolahan data menggunakan algoritme genetika dilakukan beberapa tahap untuk mendapatkan hasil yang lebih optimum, tahapan yang dilakukan yaitu :

1. Membuat populasi awal secara random lalu melakukan proses reproduksi dengan *crossover* dan mutasi pada populasi tersebut. Proses *crossover* menggunakan *one cut point crossover* dan proses mutasi menggunakan *exchange mutation*.
2. Seleksi dengan menggunakan metode *elitsm selection*.
3. Menentukan rumus *fitness* untuk memperoleh solusi yang optimal. Rumus *fitness* pada penelitian ini adalah 1000/*cost*. Dimana *cost* merupakan hasil penjumlahan dari total jarak dengan total waktu yang ditempuh oleh semua

sales.

4. Generasi dilakukan untuk memperoleh generasi berikutnya yang lebih baik.

3.4 Perancangan Sistem

Perancangan sistem merupakan tahap dimana penulis mulai merancang implementasi sistem yang digunakan untuk mendapatkan solusi optimum dalam menentukan rute optimal penjemputan penumpang *travel*. Langkah-langkah dalam proses perancangan sistem yang dilakukan pengguna yaitu memasukkan parameter berupa jumlah mobil dan parameter algoritme genetika yang akan digunakan berupa ukuran populasi (*popsize*), nilai *cr*, *mr* dan jumlah generasi. Selanjutnya sistem akan melakukan proses perhitungan menggunakan algoritme genetika yang dimulai dari inisialisasi populasi awal, proses reproduksi berupa proses *crossover* dan mutasi, proses evaluasi, serta proses seleksi. Hasil akhir yang didapatkan berupa urutan rute terbaik dalam penjemputan penumpang *travel*.

3.5 Implementasi sistem

Tahap ini mengimplementasikan hasil perancangan sistem yang akan digunakan ke dalam komputer dengan menggunakan bahasa pemrograman berorientasi objek. Implementasi pada tahap ini menggunakan menggunakan bahasa pemrograman *java* dan *software* menggunakan *Java Netbeans*. Kebutuhan pendukung dalam proses implementasi sistem akan dijelaskan lebih detail pada sub bab spesifikasi kebutuhan sistem.

3.5.1 Spesifikasi Kebutuhan Sistem

Pada tahap ini akan dilakukan analisa perangkat lunak. Tujuan dari penetapan spesifikasi kebutuhan sistem ini adalah untuk mengurangi kesalahan saat implementasi perangkat lunak. Spesifikasi perangkat yang digunakan selama pembuatan sistem optimasi penentuan rute penjemputan *travel* adalah sebagai berikut:

1. Spesifikasi *Hardware*:
    a. Laptop dengan sistem operasi Windows 10 Professional
    b. Processor Intel® Core™ i7-4500U CPU @ 1.80Ghz – 2.40Ghz
    c. RAM 4 GB
    d. 64-bit *Operating System*

2. Spesifikasi *Software*:
    a. Pemrograman menggunakan Java
    b. NetBeans IDE 8.1
    c. Browser (Google Chrome, Mozila Firefox, dll)

3.6 Analisa dan Pengujian Sistem

Pada tahap ini dilakukan pengujian berdasarkan implementasi sistem yang telah dibuat melalui perhitungan *fitness* dari alternatif solusi penentuan rute penjemputan penumpang *travel*. Pengujian dilakukan dengan cara mengganti nilai tertentu pada paramater algoritme genetika dalam setiap pengujian yang dilakukan. Pengujian ini dilakukan untuk menguji apakah metode yang digunakan telah memberikan hasil yang optimum atau belum. Selanjutnya akan dilakukan evaluasi yang bertujuan untuk mendapatkan rute optimal penjemputan penumpang *travel* .

3.7 Pengambilan Kesimpulan

Pengambilan kesimpulan merupakan sebuah proses dimana didapatkan hasil yang ditetapkan dari sebuah keputusan pilihan. Pada tahap ini berisi tentang kesimpulan dari hasil penelitian setelah dilakukannya proses perancangan, implementasi, pengujian dan analisis terhadap metode yang digunakan. Tujuan dari pengambilan kesimpulan adalah menghasilkan sebuah keputusan dari beberapa solusi alternatif. Setelah dilakukannya pengambilan kesimpulan, dilanjutkan dengan penulisan saran yang bertujuan untuk memperbaiki kesalahan pada penelitian sebelumnya sehingga terdapat sebuah pengembangan pada penelitian selanjutnya.

BAB 4 PERANCANGAN

Pada bab ini akan dibahas mengenai penerapan algoritme genetika dan perancangan sistem yang akan dibangun.

4.1 Deskripsi Masalah

Proses penjemputan penumpang *travel* sebelumnya hanya bergantung pada pengetahuan supir akan rute jalan yang dilalui, sehingga rute penjemputan penumpang *travel* menjadi kurang optimal (Samudra & Muklash, 2013). Penjemputan penumpang *travel* dilakukan oleh sopir ke alamat penumpang sesuai dengan waktu keberangkatan. Sopir akan menjemput satu persatu penumpang dan mengantarkannya ke tempat tujuan. Alamat-alamat yang akan dituju oleh sopir setiap harinya merupakan masalah yang kompleks, karena penumpang memiliki alamat yang berbeda-beda antara satu dengan yang lainnya. Penentuan rute yang kurang optimal berdampak pada biaya operasional *travel* dan waktu yang tidak effisien sehingga menyebabkan penumpang menunggu terlalu lama dan tidak tepat waktu sampai di tempat tujuan. Hal tersebut dapat menyebabkan kurangnya kualitas pelayanan yang diterima oleh penumpang. Hal ini tentunya sangat merugikan bagi pemilik jasa *travel*. Oleh sebab itu, maka diperlukan sebuah solusi yang dapat mengatasi permasalahan diatas. Salah satu metode yang digunakan untuk dapat menyelesaikan permasalahan diatas yaitu metode algoritme genetika *Multiple Travelling Salesman Problem* (M-TSP). Tujuan dari penelitian ini adalah membuat suatu sistem cerdas yang menerapkan algoritme genetika untuk mencari rute terpendek (rute optimal) dalam penjemputan penumpang *travel* dari berbagai macam alternatif solusi. Sehingga solusi baru yang dihasilkan, dapat menjadi solusi bagi perusahaan jasa *travel* dalam mengoptimpalkan proses antar jemput penumpang.

4.2 Siklus Algoritme Genetika

Di dalam penelitian ini, algoritme genetika merupakan metode yang digunakan dalam membantu mengoptimasi penentuan rute optimal penjemputan penumpang *travel*. Algoritme genetika mampu menyelesaikan sebuah permasalahan yang kompleks dan mempunyai banyak solusi alternatif (Mahmudy, 2013). Proses yang ada di dalam algoritme genetika dimulai dari tahap inisialisasi di mana tahap ini merupakan langkah awal dalam membentuk sekumpulan individu baru dengan

masing-masing individu yang dipilih secara acak. Kemudian memasuki proses reproduksi yang terdiri dari proses *crossover* dan *mutation*. Proses reproduksi bertujuan untuk menghasilkan individu baru (*children*) sebanyak *offspring* dari hasil perhitungan *crossover rate* (*cr*) dan *mutation rate* (*mr*) dengan jumlah populasi. Dari total populasi awal ditambah dengan individu baru hasil reproduksi maka tahap selanjutnya dilakukan evaluasi yang bertujuan untuk menghitung nilai *fitness* dari masing-masing individu. Nilai *fitness* berguna dalam menentukan individu mana yang memiliki kualitas terbaik. Tahap terakhir yaitu dilakukan proses seleksi yang bertujuan untuk memilih individu-individu baru dari sekumpulan populasi dan *offspring* guna menentukan individu mana yang akan ditetapkan sebagai penerus generasi berikutnya sesuai dengan nilai *fitness* yang dihasilkan. Semakin besar nilai *fitness* yang dihasilkan maka semakin besar pula kesempatan individu tersebut terpilih (Mawaddah & Mahmudy, 2006). Semua proses tersebut akan terus berulang sebanyak total iterasi yang telah ditentukan sebelumnya.

4.3 Siklus Penyelesaian Masalah Menggunakan Algoritme Genetika

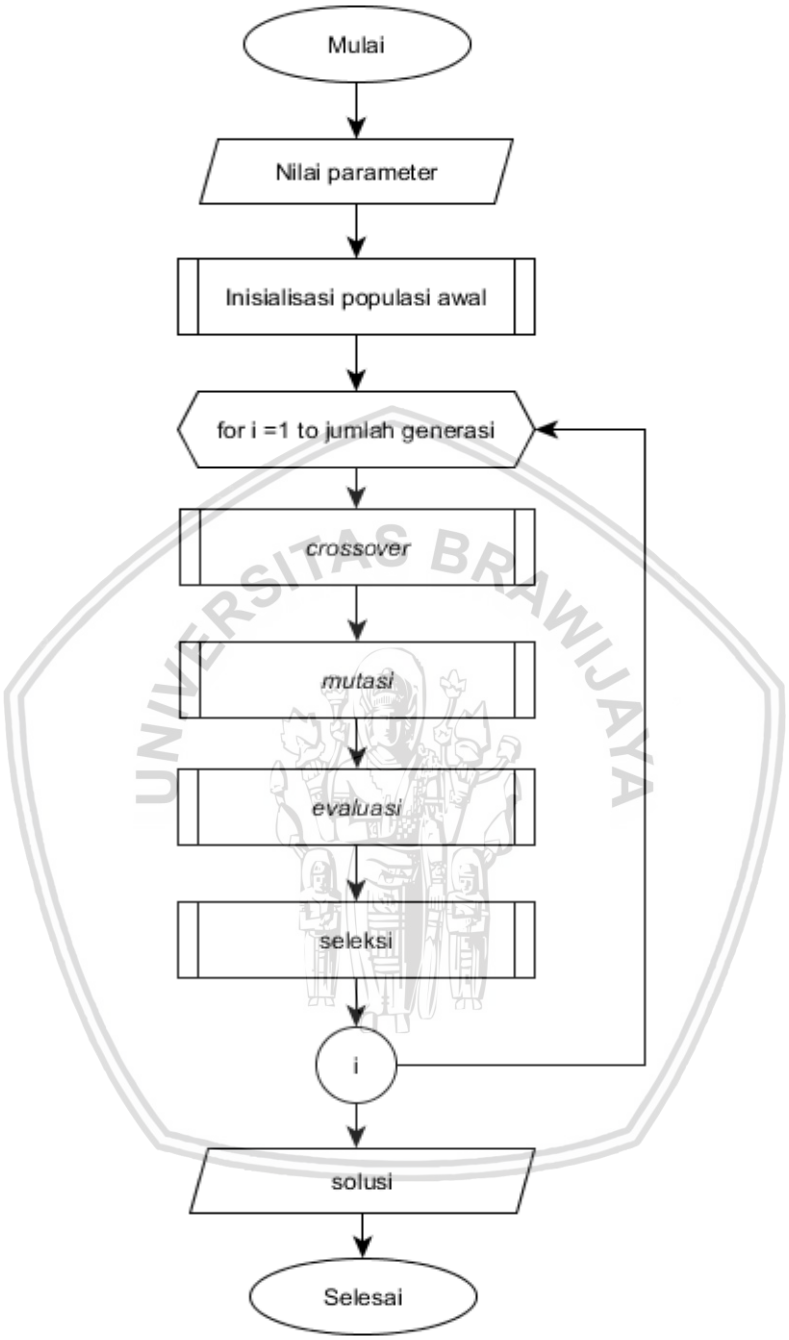
Pemecahan masalah pada skripsi ini menggunakan metode algoritme genetika dikarenakan pada algoritme genetika mempunyai kemampuan untuk memecahkan masalah yang komplek dan mempunyai banyak solusi alternatif (Mahmudy,2013). Cara kerja algoritme genetika pada permasalahan ini yaitu dengan memilih satu solusi yang terbaik. Langkah-Langkah penerapan algoritme genetika untuk penyelesaian masalah penentuan rute penjemputan penumpang *travel* yaitu:

1. Proses insialisasi parameter dengan memasukkan parameter algoritme genetika yaitu :
    a. Jumlah populasi (*popsize*).
    b. *Crossover rate* (*Cr*).
    c. *Mutation rate* (*Mr*).
    d. Jumlah generasi atau iterasi.
2. Inisialisasi populasi awal berdasarkan *popsize* yang telah ditentukan.
3. Pembentukan populasi baru dengan proses reproduksi untuk generasi selanjutnya. Pada tahap ini proses reproduksi meliputi :

a) Proses *crossover* menggunakan metode *one cut point crossover*. Banyak *offspring* pada proses *crossover* ditentukan oleh perhitungan dari *crossover rate* (*cr*) dikali dengan *popsize*.
b) Proses mutasi pada penelitian ini, menggunakan *reciprocal exchange mutation*. Banyak *offspring* pada proses mutasi ditentukan oleh perhitungan dari *mutation rate* (*mr*) dikali dengan *popsize*.
c) Menghitung nilai *fitness* pada masing-masing kromosom.
d) Melakukan proses seleksi dengan menggunakan metode *elitsm* untuk memilih individu-individu baru dari induk dan anak hasil reproduksi dengan merangking berdasarkan nilai *fitness* tertinggi. Pemilihan individu baru

disesuaikan dengan besaran populasi yang telah ditentukan. Hasil seleksi akan dijadikan sebagai populasi pada generasi berikutnya.

4. Apabila kondisi akhir telah memenuhi kriteria, maka dilanjutkan dengan pemilihan kromosom dengan nilai *fitness* tertinggi sebagai solusi terbaik. Jika kondisi pada iterasi belum mencapai akhir maka dilanjutkan kembali ke iterasi selanjutnya. Secara umum proses urutan penyelesaian masalah penentuan rute penjemputan penumpang *travel* menggunakan algoritme genetika ditunjukkan pada Gambar 4.1.



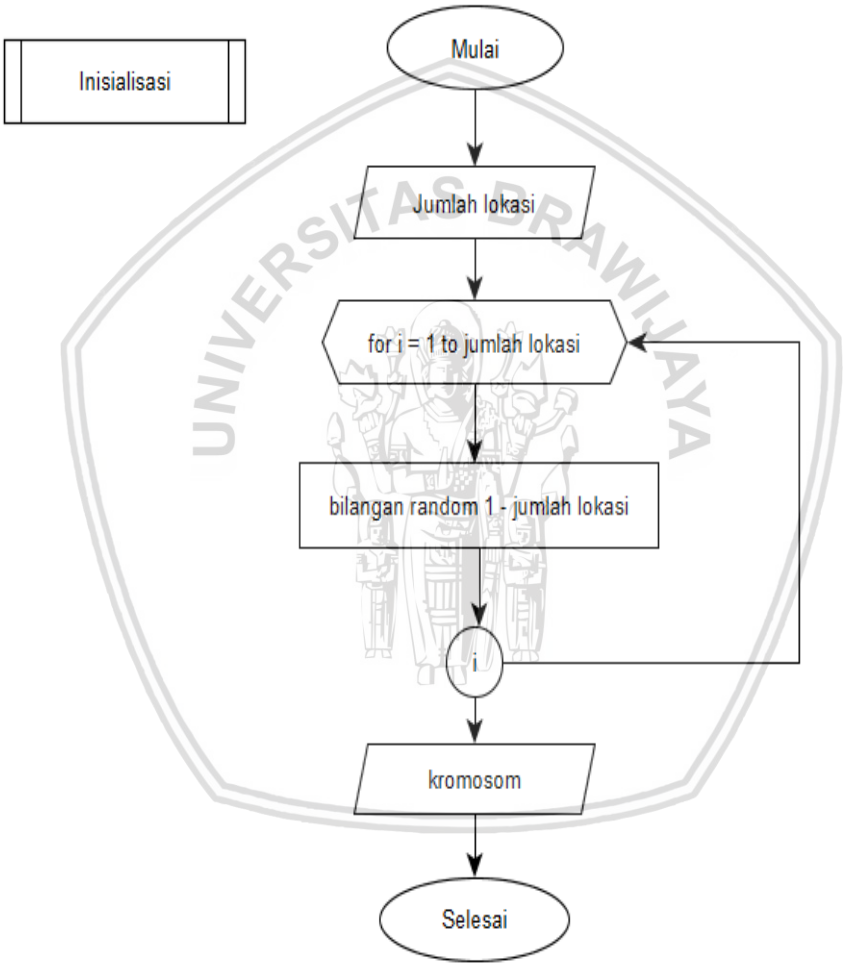
Gambar 4.1 *Flowchart* Proses Implementasi Algoritme Genetika

Pada Gambar 4.1 merupakan *flowchart* yang menggambarkan keseluruhan proses dari algoritme genetika. Proses diawali dengan memasukkan nilai parameter berupa nilai *cr, mr* dan jumlah generasi. Kemudian dilanjutkan dengan melakukan proses inisialisasi populasi awal yang bertujuan untuk membangkitkan himpunan

solusi baru secara *random* ke dalam sebuah populasi. Selanjutnya proses reproduksi yang terdiri dari dua proses yaitu proses *crossover* dan mutasi. Setelah melakukan proses reproduksi, dilanjutkan dengan proses evaluasi.

4.3.1 Inisialisasi Populasi Awal

Inisialisasi populasi awal merupakan tahapan untuk menentukan jumlah kromosom yang akan digunakan. Pada penelitian ini, panjang kromosom tergantung dari banyaknya daerah yang akan dikunjungi. Proses dari inisialisasi populasi awal dapat dilihat pada Gambar 4.2.

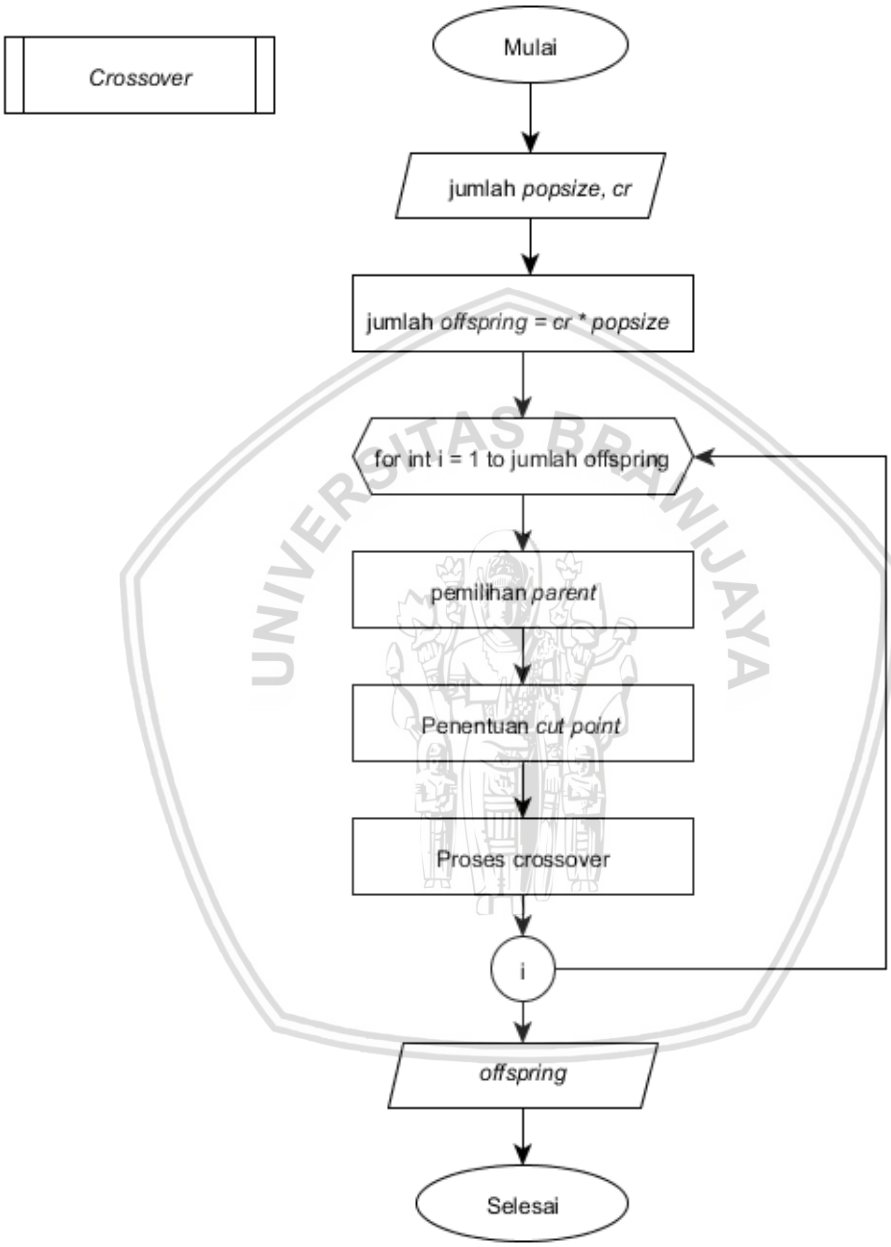


Gambar 4.2 *Flowchart* Proses Inisialisasi Populasi Awal

4.3.2 Reproduksi

Setelah terbentuknya populasi, dilanjutkan dengan proses reproduksi yang bertujuan untuk menghasilkan ketuturunan berikutnya. Pada penelitian ini, proses reproduksi dilakukan dengan dua cara, yaitu pindah silang (*crossover*) dan mutasi (*mutation*). Hasil dari reproduksi akan menghasilkan keturunan (*offspring*).

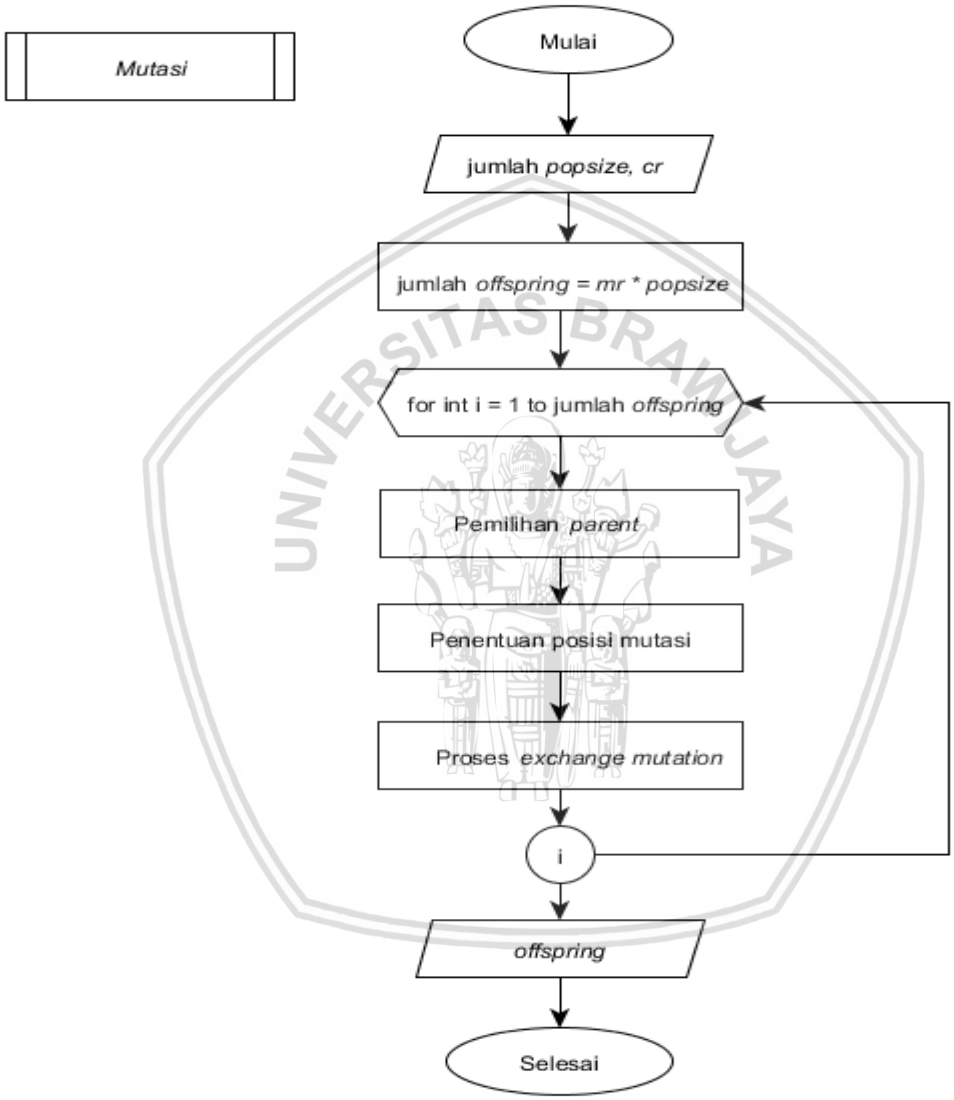
Flowchart proses *crossover* dapat dilihat pada Gambar 4.2 dan *flowchart* dari proses mutasi dapat dilihat pada Gambar 4.3.



Gambar 4.3 *Flowchart* Proses *Crossover*

Gambar 4.3 menjelaskan tentang proses *crossover* yang diawali dengan memasukkan nilai *cr* dan jumlah *popsize* untuk menentukan jumlah *offspring,* setelah itu dilakukan pemilihan parent secara random yang akan digunakan dalam proses *crossover.* Kemudian dilanjutkan dengan penentuan posisi *cutpoint* menggunakan metode *one cut point crossover* dengan memilih satu titik potong

kromosom. Selanjutnya dilakukan proses *crossover* dengan cara memotong kromosom *parent* 1 (P1) pada *cut point* yang telah ditentukan, kemudian menggabungkannya dengan potongan kromosom *parent* 2 (P2) sehingga didapatkan *child* (C1) dan sebaliknya memotong kromosom P2 pada *cut point* yang telah ditentukan, kemudian menggabungkannya dengan potongan kromosom P1 sehingga didapatkan *child* (C2).



Gambar 4.4 *Flowchart* Proses Mutasi

Gambar 4.4 menjelaskan tentang proses *mutasi* yang diawali dengan memasukkan nilai *mr* dan jumlah *popsize* untuk menentukan jumlah *offspring,* setelah itu dilakukan pemilihan *parent* yang akan digunakan dalam proses *mutasi.* Kemudian dilanjutkan dengan penentuan posisi *mutasi* menggunakan metode *exchange mutation* dengan memilih dua titik secara acak. Setelah dipilih dua titik

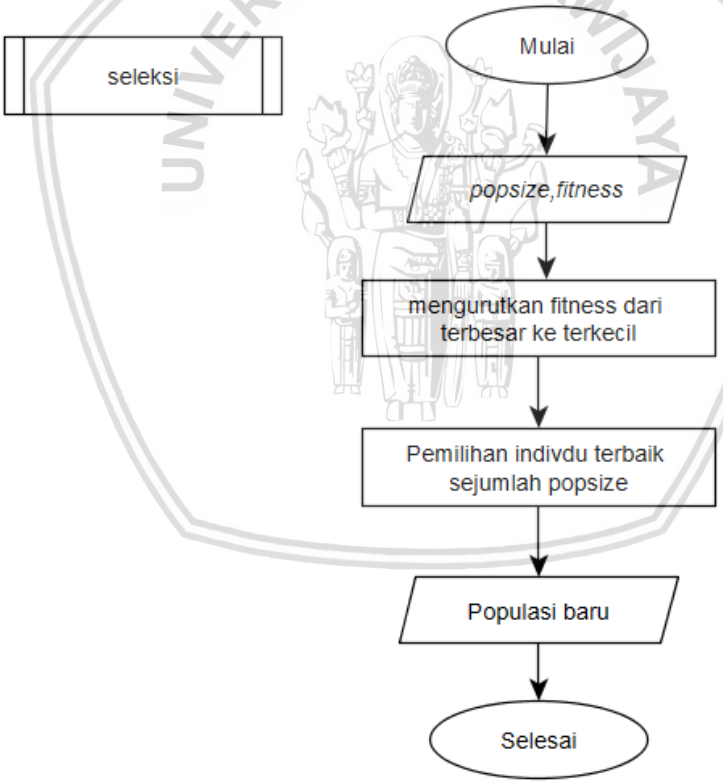
secara acak, selanjutnya akan dilakukan penukaran dua titik tersebut dan banyak daerah yang dikunjungi oleh *salesman* akan ditukar.

4.3.3 Evaluasi

Proses *evaluasi* merupakan proses pengumpulan semua individu dalam populasi beserta dengan *child*nya kemudian dihitung *fitness* dari masing-masing kromosom. Perhitungan dari nilai *fitness* merupakan perhitungan untuk mengetahui apakah indivitu tersebut sudah baik jika dijadikan sebuah solusi.

4.3.4 Seleksi

Proses seleksi merupakan proses untuk menyeleksi individu mana saja yang lolos untuk menjadi populasi pada generasi selanjutnya. Penelitian ini menggunakan metode *elitism selection*. Metode ini melakukan penyeleksian dengan mengurutkan individu berdasarkan nilai *fitness* terbesar ke terkecil sebanyak jumlah populasi. Proses dari seleksi dapat dilihat pada Gambar 4.5.



Gambar 4.5 Flowchart Proses Seleksi

4.4 Perhitungan Manual

Perhitungan manual optimasi penentuan rute optimal penjemputan penumpang *travel* menggunakan beberapa parameter yang dibutuhkan seperti jumlah salesman, ukuran populasi, *crossover rate, mutation rate*, jumlah generasi serta angka permutasi yang akan menentukan banyak daerah yang nantinya akan dikunjungi dalam setiap populasi. Pada perhitungan manualisasi ini menggunakan sampel dari beberapa parameter tersebut, diantaranya adalah :

Banyak *Salesman* : 2 *salesman*
Ukuran Populasi (*popsize*) : 10
Crossover Rate (*Cr*) : 0,6
Mutation Rate (*Mr*) : 0,4
Jumlah Generasi : 1
Angka permutasi : [1-10]

4.4.1 Data Manualisasi

Dalam penelitian ini data yang digunakan berupa data jarak dan waktu tempuh perjalanan. Pada perhitungan manual (manualisasi) menggunakan sampel data jarak dan waktu sebanyak 10 lokasi dengan menggunakan 2 orang salesman. Data sebelum dilakukan normalisasi dapat dilihat pada tabel 4.1 dan 4.2.

Tabel 4.1 Data Jarak Antar Penumpang

Node	Pangkalan	1	2	3	4	5	6	7	8	9	10
Pangkalan	0	1,5	8,0	6,0	6,0	5,5	6,8	4,7	4,5	1,8	6,6
1	1,5	0	8,9	4,9	5,3	4,2	6,9	3,0	3,1	1,8	5,2
2	8,0	8,9	0	13,7	11,7	11,9	11,0	11,7	12,3	8,9	13,1
3	6,0	4,9	13,7	0	5,2	3,6	6,7	3,4	4,0	5,2	4,1
4	6,0	5,3	11,7	5,2	0	1,5	1,8	3,3	3,6	6,9	2,2
5	5,5	4,2	11,9	3,6	1,5	0	4,0	1,6	2,3	5,8	1,0
6	6,8	6,9	11,0	6,7	1,8	4,0	0	5,4	5,2	8,4	4,3
7	4,7	3,0	11,7	3,4	3,3	1,6	5,4	0	0,7	3,5	2,1
8	4,5	3,1	12,3	4,0	3,6	2,3	5,2	0,7	0	4,6	3,5
9	1,8	1,8	8,9	5,2	6,9	5,8	8,4	3,5	4,6	0	5,7
10	6,6	5,2	13,1	4,1	2,1	1,0	4,3	2,1	3,5	5,7	0

Tabel 4.2 Waktu Perjalanan

Data waktu (jam)											
Waktu	Pangkalan	1	2	3	4	5	6	7	8	9	10
Pangkalan	0,0	0,1	0,3	0,3	0,3	0,2	0,3	0,2	0,2	0,1	0,3

1	0,1	0,0	0,6	0,3	0,3	0,3	0,4	0,3	0,2	0,1	0,4
2	0,3	0,6	0,0	0,9	0,7	0,8	0,7	0,7	0,7	0,4	0,8
3	0,3	0,3	0,9	0,0	0,3	0,2	0,4	0,2	0,2	0,4	0,2
4	0,3	0,3	0,7	0,3	0,0	0,3	0,1	0,2	0,2	0,4	0,1
5	0,2	0,3	0,8	0,2	0,3	0,0	0,2	0,1	0,1	0,4	0,2
6	0,3	0,4	0,7	0,4	0,1	0,2	0,0	0,2	0,2	0,4	0,2
7	0,2	0,3	0,7	0,2	0,2	0,1	0,2	0,0	0,2	0,2	0,1
8	0,2	0,2	0,7	0,2	0,2	0,1	0,2	0,2	0,0	0,2	0,2
9	0,1	0,1	0,4	0,4	0,4	0,4	0,4	0,2	0,2	0,0	0,3
10	0,3	0,4	0,8	0,2	0,1	0,2	0,2	0,1	0,2	0,3	0,0

Tabel 4.3 Daftar Lokasi Penjemputan penumpang *travel*

No	Lokasi
1	Jl. Pisang Kipas No.10
2	Perum. Banjararum Asri Blok B no.21
3	Perum. Bukit Cemara Tidar Blok B no.33
4	Jl. Bareng Tengah V no.681
5	Jl. Klampok Kasri 2 no.231
6	Jl. Pasar Besar no.58
7	Jl. Bendungan Sutami no.2B
8	Jl.Veteran no.2
9	Jl Candi Panggung Barat
10	Jl. Halimun no.8, Dieng

Perhitungan normalisasi data menggunakan rumus *min-max* normalisasi. Data jarak dan waktu dipetakan dari range 1 – 10 sehingga data jarak dan waktu tidak memiliki perbedaan nilai yang terlalu jauh. Contoh perhitungan normalisasi data jarak adalah sebagai berikut :

N' = (N – Min) / (Max-Min) * (New_Max – New_Min) + New_Min
N12' = (N12 – 0,4) / (17,4 – 0,4) * (10 – 1) + 1
N12' = (8,9 – 0,4) / (17,4 – 0,4) * (10 – 1) + 1
N12' = 5,50

Sedangkan normalisasi dari data waktu dapat dihitung dengan:

N' = (N – Min) / (Max-Min) * (New_Max – New_Min) + New_Min
N12' = (N12 – 0,4) / (17,4 – 0,4) * (10 – 1) + 1
N12' = (0,57 – 0,03) / (0,93 – 0,03) * (10 – 1) + 1
N12' = 6,4

Setelah dilakukan proses normalisasi data maka akan terjadi perubahan nilai pada data sesuai dengan range yang ditentukan berdasarkan rumus normalisasi min-

max. Data setelah proses normalisasi dapat dilihat pada tabel 4.7

Tabel 4.4 Data Jarak Setelah Normalisasi

Node	pangkalan	1	2	3	4	5	6	7	8	9	10
pangkalan	0	1,58	5,02	3,96	3,96	3,70	4,39	3,28	3,17	1,74	4,28
1	1,58	0	5,50	3,38	3,59	3,01	4,44	2,38	2,43	1,74	3,54
2	5,02	5,50	0	8,04	6,98	7,09	6,61	6,98	7,30	5,50	7,72
3	3,96	3,38	8,04	0	3,54	2,69	4,34	2,59	2,91	3,54	2,96
4	3,96	3,59	6,98	3,54	0	1,58	1,74	2,54	2,69	4,44	1,95
5	3,70	3,01	7,09	2,69	1,58	0	2,91	1,64	2,01	3,86	1,32
6	4,39	4,44	6,61	4,34	1,74	2,91	0	3,65	3,54	5,24	3,06
7	3,28	2,38	6,98	2,59	2,54	1,64	3,65	0	1,16	2,64	1,90
8	3,17	2,43	7,30	2,91	2,69	2,01	3,54	1,16	0	3,22	2,64
9	1,74	1,74	5,50	3,54	4,44	3,86	5,24	2,64	3,22	0	3,81
10	4,28	3,54	7,72	2,96	1,90	1,32	3,06	1,90	2,64	3,81	0

Tabel 4.5 Data Waktu Setelah Normalisasi

Node	pangkalan	1	2	3	4	5	6	7	8	9	10
pangkalan	0	1,7	4,0	4,0	3,2	3,0	3,4	3,0	3,0	1,5	3,7
1	1,7	0	6,4	3,5	3,4	3,7	4,7	3,4	3,0	1,7	4,4
2	4,0	6,4	0	10	7,9	8,2	7,4	7,5	7,7	4,9	8,5
3	4,0	3,5	10	0	3,5	2,5	4,2	2,7	3	4,7	2,9
4	3,2	3,4	7,9	3,5	0	3,9	1,9	2,4	2,4	4,7	2,0
5	3,0	3,7	8,2	2,5	3,9	0	2,5	1,7	2	4,5	2,7
6	3,4	4,7	7,4	4,2	1,9	2,5	0	3,0	3,0	4,2	2,7
7	3,0	3,4	7,5	2,7	2,4	1,7	3,0	0	2,2	3,0	1,7
8	3,0	3,0	7,7	3,0	2,4	2,0	3,0	2,2	0	2,9	2,4
9	1,5	1,7	4,9	4,7	4,7	4,5	4,2	3,0	2,9	0	3,5
10	3,7	4,4	8,5	2,9	2	2,7	2,7	1,7	2,4	3,5	0

4.4.2 Representasi Kromosom

Untuk mendifinisikan setiap individu dalam populasi diperlukan suatu proses pengkodean kromosom atau disebut dengan representasi kromosom. Pada penelitian ini menggunakan representasi kromosom permutasi dengan nilai gen merupakan bilangan permutasi pada interval nilai dari 1 sampai dengan 10 yang merupakan sampel dari data lokasi penjemputan penumpang. Panjang kromosom tergantung dari banyaknya lokasi penjemputan penumpang yang akan dikunjungi dimana pada perhitungan manual ini mengambil sampel dengan menggunakan dua orang salesman. Contoh populasi awal yang dibangkitkan dapat dilihat pada tabel 4.6.

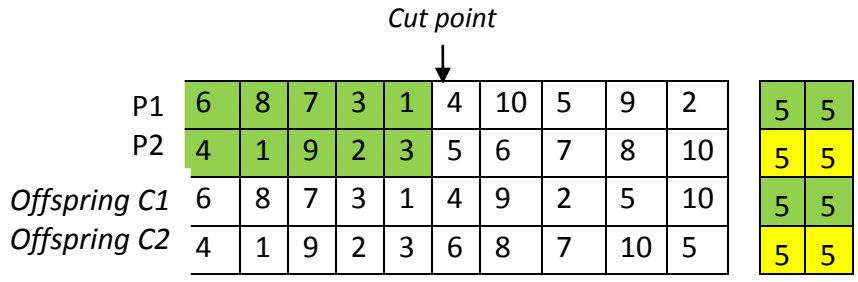
Tabel 4.6 Populasi awal

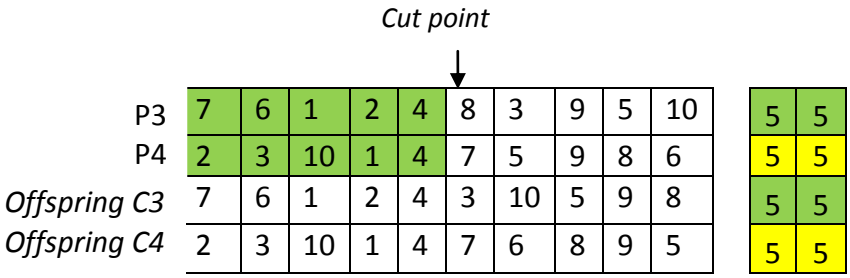
Individu	1	2	3	4	5	6	7	8	9	10	11	12
P1	6	10	8	9	6	5	1	3	4	2	5	5
P2	8	7	5	3	10	1	4	6	2	9	5	5
P3	6	8	7	3	1	4	10	5	9	2	5	5
P4	7	6	1	2	4	8	3	9	5	10	5	5
P5	5	4	2	6	7	1	9	10	3	8	5	5
P6	4	1	9	2	3	5	6	7	8	10	5	5
P7	3	5	4	6	7	2	8	10	9	1	5	5
P8	2	3	10	1	4	7	5	9	8	6	5	5
P9	9	10	8	3	2	1	7	4	6	5	5	5
P10	10	9	6	1	5	3	2	8	7	4	5	5
	Segmen 1										Segmen 2	

Segmen 1 pada tabel 4.6 menunjukkan banyaknya lokasi penjemputan penumpang dan segmen 2 menunjukkan banyak penumpang yang dijemput setiap *salesman,* dengan asumsi bahwa setiap *salesman* berangkat dari kantor *travel*.

4.4.3 Crossover

Pada penelitian ini, metode *crossover* yang digunakan adalah *one cut point crossover*. Metode o*ne cut point crossover* dilakukan dengan cara memilih dua induk secara random, kemudian pada kedua induk tersebut akan dipilih satu titik gen yang akan dijadikan titik potong kromosom pada proses tukar silangnya. Proses tukar silang dilakukan dengan cara mempasangkan panjang gen sebelah kiri dari induk pertama dengan panjang gen sebelah kanan dari induk ke dua. Pada segmen 1 menunjukan jumlah lokasi penjemputan penumpang sebanyak 10 dan pada segmen 2 menunjukkan banyaknya penumpang yang dijemput setiap *salesman.* Nilai P1 pada segmen kedua akan diwariskan kepada C1 dan nilai P2 pada segmen kedua akan diwariskan kepada C2. Dari hasil proses tukar silang tersebut akan dihasilkan individu baru atau *child.* Pada penelitian ini menggunakan *crossover rate* dengan nilai Cr = 0.4 dan banyaknya populasi (*popsize)* = 10, maka akan dihasilkan 4 *child* pada proses *crossover* karena populasi (*popsize*) = 10 * 0.4 = 4, keturunan yang dihasilkan dapat dilihat pada gambar 4.3.

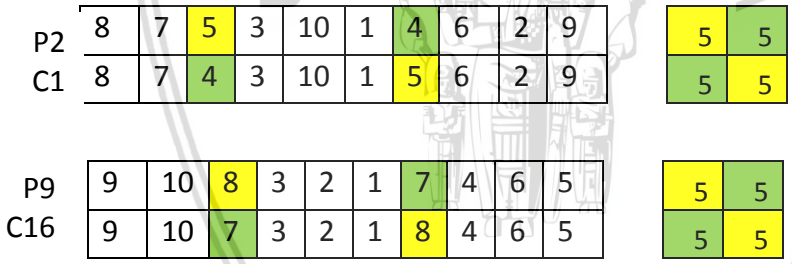




Gambar 4.3 Proses Crossover

4.4.4 Mutasi

Proses reproduksi selanjutnya adalah mutasi. Pada penelitian ini, metode mutasi yang digunakan adalah *reciprocal exchange mutation*. Metode *exchange mutation* dilakukan dengan cara memilih dua titik gen secara acak antara titik 1 dan titik 2, kemudian antara kedua titik gen tersebut akan ditukarkan posisinya. Pada penelitian ini menggunakan *mutation rate* dengan nilai Mr = 0.2 dan banyaknya populasi (*popsize)*=10, maka akan dilakukan 2 kali proses *mutasi* karena populasi (*popsize*) = 10*0.2 = 2, Contoh proses mutasi dapat dilihat pada gambar 4.4.



Gambar 4.4 Proses Mutasi

4.4.5 Evaluasi

Proses evaluasi dilakukan untuk menghitung nilai *fitness* dari masing-masing individu yang ada di dalam populasi. Dalam perhitungan nilai *fitness* terdapat parameter *cost*, dimana *cost* merupakan hasil dari total jarak (km) ditambah dengan total waktu (jam). Setelah didapatkan *cost* dari masing-masing individu, maka dapat dilakukan perhitungan *fitness* dengan menggunakan rumus :

$$Fitness = \frac{1000}{cost}$$

Cost = total jarak + total waktu

Parameter yang dihitung dalam penelitian ini adalah total jarak dan total

waktu yang digunakan untuk mencari nilai *cost*, dimana perhitungan total jarak dan total waktu dimulai dari pangkalan *travel*. Dalam perhitungan manualiasi menggunakan 2 *salesman* atau 2 mobil untuk penjemputan 10 orang penumpang *travel*. Contoh perhitungan total jarak dan total waktu adalah sebagai berikut :

Kromosom : [1 2 3 4 5 6 7 8 9 10]
Jalur Mobil 1 = [1 2 3 4 5]
= 1,58 + 5,50 + 8,04 + 3,54 + 1,58 = 20,25
Jalur Mobil 2 = [6 7 8 9 10]
= 4,39 + 3,65 + 1,16 + 3,22 + 3,81 = 16,22
Total Jarak = 20,25 + 16,22 = 36,47

Kromosom : [1 2 3 4 5 6 7 8 9 10]
Waktu Mobil 1 = [1 2 3 4 5]
= 1,7 + 6,4 + 10,0 + 3,5 + 3,9 = 25,50
Waktu Mobil 2 = [6 7 8 9 10]
= 3,4 + 3,0 + 2,2 + 2,9 + 3,5 = 15,00
Total Jarak = 25,50 + 15,00 = 40,50

Cost = Total Jarak + Total Waktu
= 36,47 + 40,50 = 76,97

Fitness = $\frac{1000}{cost} = \frac{1000}{76,97} = 1,299$

Tabel 4.7 Proses Evaluasi

Individu	Kromosom	Segmen Mobil	Total Jarak	Total Waktu	*Cost*	*Fitness*
P1	[1 2 3 4 5 6 7 8 9 10]	[5 5]	66,1	3,88	69,98	1,429
P2	[8 7 5 3 10 1 4 6 2 9]	[5 5]	51,4	2,8	54,24	1,844
P3	[6 8 7 3 1 4 10 5 9 2]	[5 5]	54,4	2,9	57,34	1,744
P4	[7 6 1 2 4 8 3 9 5 10]	[5 5]	70,7	4,1	74,84	1,336
P5	[5 4 2 6 7 1 9 10 3 8]	[5 5]	61,4	3,6	64,96	1,539
P6	[4 1 9 2 3 5 6 7 8 10]	[5 5]	67,4	3,6	70,96	1,409
P7	[3 5 4 6 7 2 8 10 9 1]	[5 5]	51,2	2,9	54,06	1,850
P8	[2 3 10 1 4 7 5 9 8 6]	[5 5]	71,0	3,8	74,80	1,337
P9	[9 10 8 3 2 1 7 4 6 5]	[5 5]	55,8	3,1	58,89	1,698
P10	[10 9 6 1 5 3 2 8 7 4]	[5 5]	79,3	4,4	83,69	1,195
C11	[3 5 4 6 7 2 8 1 10 9]	[5 5]	59,1	3,4	62,50	1,600
C12	[8 7 5 3 10 1 4 6 2 9]	[5 5]	51,4	2,84	54,24	1,844
C13	[1 2 5 4 6 7 9 10 3 8]	[5 5]	58,9	3,55	62,45	1,601

C14	[5 4 1 2 3 6 7 8 9 10]	[5 5]	70,7	4,1	74,80	1,337
C15	[8 7 5 3 4 1 10 6 2 9]	[5 5]	54,3	3,03	57,33	1,744
C16	[7 5 4 6 3 2 8 8 9 1]	[5 5]	50,5	2,9	53,40	1,873
C17	[7 5 1 2 4 8 3 9 6 10]	[5 5]	70,1	3,88	73,98	1,352
C18	[8 4 2 6 7 1 9 10 3 5]	[5 5]	63,1	3,36	66,46	1,505
C19	[1 3 10 2 4 7 5 9 8 6]	[5 5]	70	3,78	73,78	1,355
C20	**[6 9 7 3 1 4 10 5 8 2]**	**[5 5]**	**60,3**	**3,17**	**63,47**	**1,576**

4.4.5 Seleksi

Pada tahap ini merupakan proses untuk menyeleksi individu mana saja yang lolos untuk menjadi populasi pada generasi selanjutnya. Penelitian ini menggunakan metode *elitism selection*. Metode ini melakukan penyeleksian dengan mengurutkan individu berdasarkan nilai *fitness* terbesar ke terkecil sebanyak jumlah populasi (*popsize*) yang telah ditentukan. Tahapan dari proses seleksi adalah sebagai berikut:

1. Mengurutkan individu *parent* dan *child* menggunakan nilai *fitness* dari nilai yang terbesar sampai terkecil sesuai dengan hasil proses evaluasi.

Tabel 4.8 Proses Seleksi

Individu	Kromosom	Segmen Mobil	*Fitness*
P1	[1 2 3 4 5 6 7 8 9 10]	[5 5]	1,429
P2	[8 7 5 3 10 1 4 6 2 9]	[5 5]	1,844
P3	[6 8 7 3 1 4 10 5 9 2]	[5 5]	1,744
P4	[7 6 1 2 4 8 3 9 5 10]	[5 5]	1,336
P5	[5 4 2 6 7 1 9 10 3 8]	[5 5]	1,539
P6	[4 1 9 2 3 5 6 7 8 10]	[5 5]	1,409
P7	[3 5 4 6 7 2 8 10 9 1]	[5 5]	1,850
P8	[2 3 10 1 4 7 5 9 8 6]	[5 5]	1,337
P9	[9 10 8 3 2 1 7 4 6 5]	[5 5]	1,698
P10	[10 9 6 1 5 3 2 8 7 4]	[5 5]	1,195
C11	[3 5 4 6 7 2 8 1 10 9]	[5 5]	1,600
C12	[8 7 5 3 10 1 4 6 2 9]	[5 5]	1,844
C13	[1 2 5 4 6 7 9 10 3 8]	[5 5]	1,601
C14	[5 4 1 2 3 6 7 8 9 10]	[5 5]	1,337
C15	[8 7 5 3 4 1 10 6 2 9]	[5 5]	1,744
C16	[7 5 4 6 3 2 8 8 9 1]	[5 5]	1,873
C17	[7 5 1 2 4 8 3 9 6 10]	[5 5]	1,352
C18	[8 4 2 6 7 1 9 10 3 5]	[5 5]	1,505
C19	[1 3 10 2 4 7 5 9 8 6]	[5 5]	1,355
C20	[6 9 7 3 1 4 10 5 8 2]	[5 5]	1,576

2. Memilih individu terbaik menggunakan metode seleksi *elitism* dengan mengurutkan nilai *fitness* terbesar ke terkecil berdasarkan jumlah *popSize* dan selanjutnya individu yang terpilih akan dijadikan populasi baru pada generasi berikutnya. Sehingga dari 20 individu yang ada pada populasi awal, hanya 10 individu yang bisa masuk menjadi populasi generasi berikutnya. Hasil seleksi *elitism* ditunjukkan pada Tabel 4.9.

Tabel 4.9 Hasil Seleksi *Elitism*

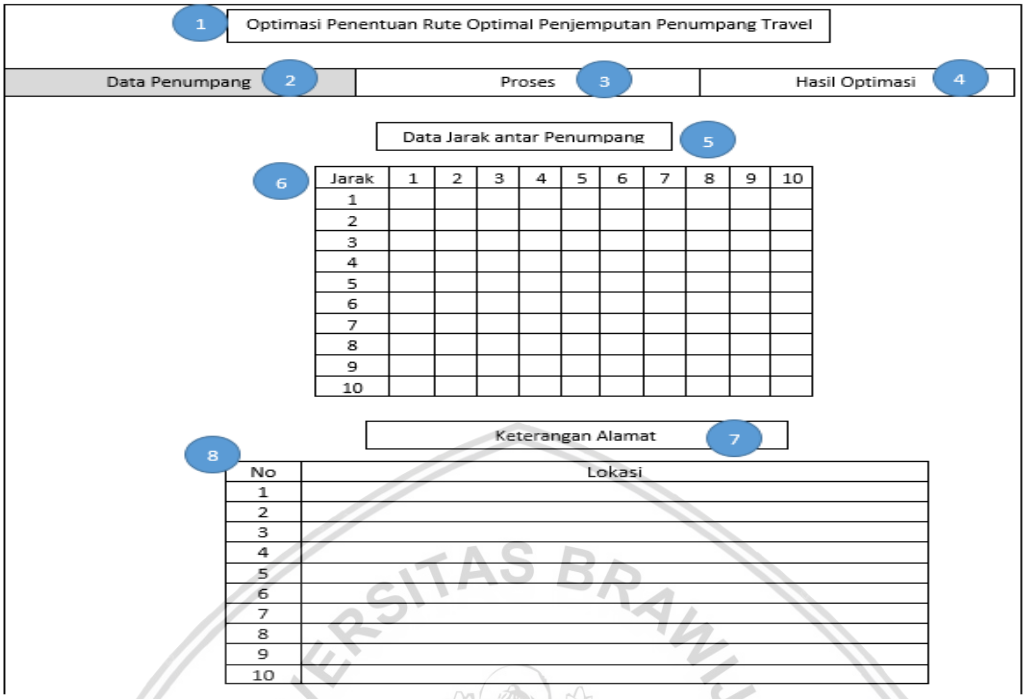
Individu	Kromosom	Segmen Mobil	*Fitness*
P1	[7 5 4 6 3 2 8 8 9 1]	[5 5]	1,873
P2	[3 5 4 6 7 2 8 10 9 1]	[5 5]	1,850
P3	[8 7 5 3 10 1 4 6 2 9]	[5 5]	1,844
P4	[8 7 5 3 10 1 4 6 2 9]	[5 5]	1,844
P5	[8 7 5 3 4 1 10 6 2 9]	[5 5]	1,744
P6	[6 8 7 3 1 4 10 5 9 2]	[5 5]	1,744
P7	[9 10 8 3 2 1 7 4 6 5]	[5 5]	1,698
P8	[1 2 5 4 6 7 9 10 3 8]	[5 5]	1,601
P9	[3 5 4 6 7 2 8 1 10 9]	[5 5]	1,600
P10	[5 4 2 6 7 1 9 10 3 8]	[5 5]	1,539

4.5 Perancangan User Interface

Pada perancangan *user interface* dari perangkat lunak untuk permasalahan penentuan rute optimpal penjemputan penumpang *travel* ini terdiri dari 3 tampilan. Tampilan yang tersedia terdiri dari halaman data penumpang, halaman proses dan *input* parameter, serta halaman hasil optimasi.

4.4.1 Halaman Data Penumpang

Halaman data penumpang merupakan halaman yang menampilkan data data jarak antar penumpang dan keterangan alamat. Halaman data penumpang ditunjukkan pada Gambar 4.5.



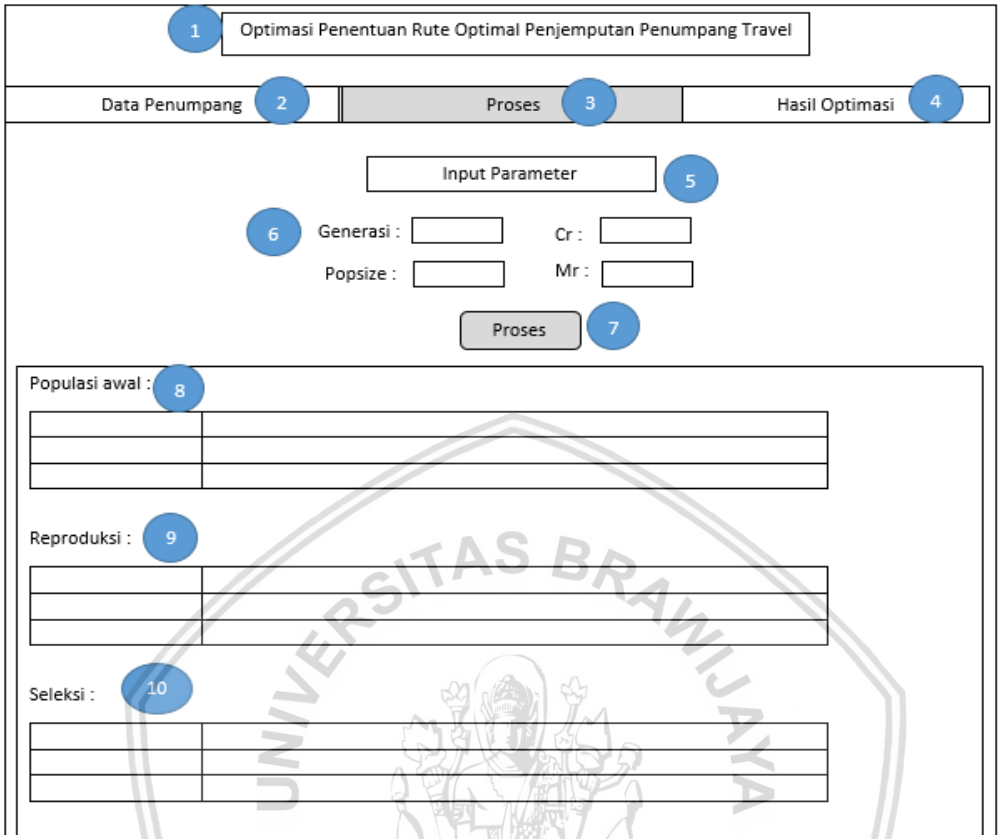
Gambar 4.5 Halaman data penumpang

Berikut keterangan dari halaman data penumpang:

1. Judul program/aplikasi
2. *Sub menu* halaman data penumpang
3. *Sub menu* halaman proses dan *input* parameter
4. *Sub menu* halaman hasil optimasi
5. Judul tabel
6. Tabel data jarak antar penumpang
7. Judul tabel
8. Tabel keterangan alamat

4.4.2 Halaman Proses dan *Input* Parameter

Halaman proses dan input parameter merupakan halaman yang berisikan kolom masukkan parameter dan tabel hasil proses algoritme genetika. Halaman proses dan *input* parameter ditunjukkan pada Gambar 4.6.



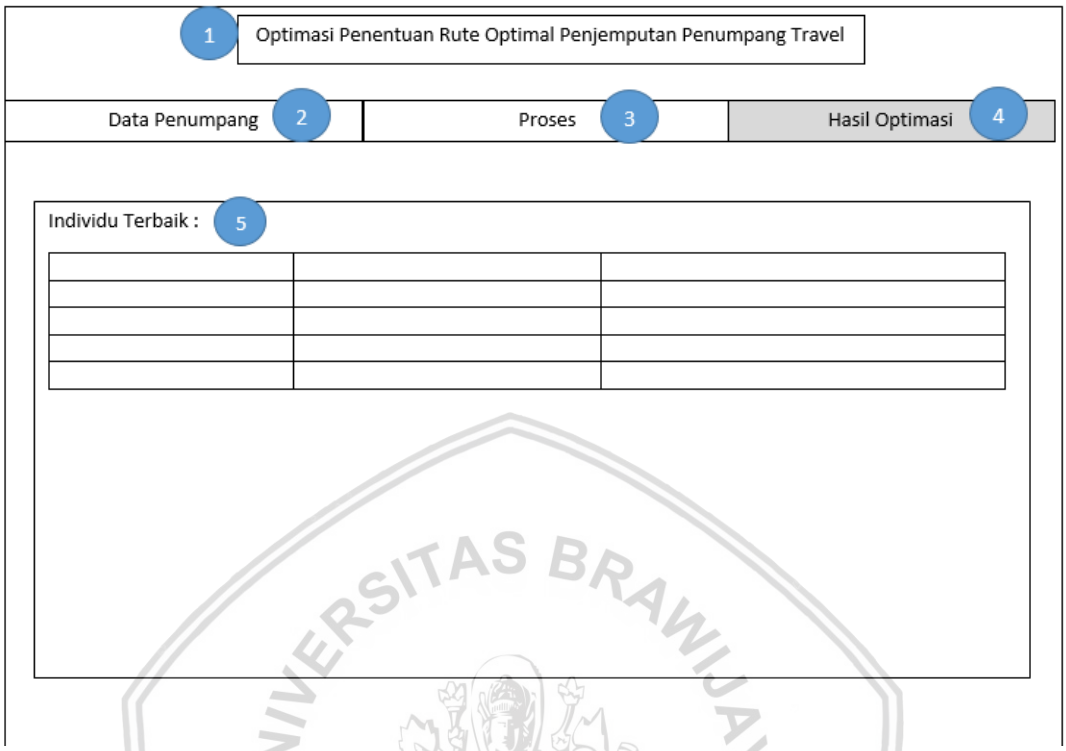
Gambar 4.6 Halaman Proses dan Input Parameter

Berikut keterangan dari halaman data petugas:

1. Judul program/aplikasi
2. *Sub menu* halaman data penumpang
3. *Sub menu* halaman proses dan *input* parameter
4. *Sub menu* halaman hasil optimasi
5. Judul tabel
6. Form *input* parameter (generasi, populasi, *cr* dan *mr*)
7. Tombol proses
8. Judul dan tabel hasil pembentukan populasi awal
9. Judul dan tabel hasil reproduksi
10. Judul dan tabel hasil seleksi

4.4.3 Halaman Hasil Optimasi

Halaman hasil optimasi merupakan halaman yang menampilkan tabel pemilihan individu terbaik. Halaman hasil optimasi ditunjukkan pada Gambar 4.7.



Gambar 4.7 Halaman Hasil Optimasi

Berikut keterangan dari halaman hasil optimasi:
1. Judul program/aplikasi
2. *Sub menu* halaman data penumpang
3. *Sub menu* halaman proses dan *input* parameter
4. *Sub menu* halaman hasil optimasi
5. Judul dan tabel pemilihan individu terbaik

4.5 Perancangan Pengujian

Proses perancangan skenario uji coba yang dilakukan pada penelitian ini menggunakan 3 skenario pengujian. Uji coba ini dilakukan untuk mendapatkan solusi atau hasil dari pemecahan masalahan yang lebih optimal. Tiap skenario akan dilakukan pengujian sebanyak 10 kali dan selanjutnya akan diambil rata-rata nilai *fitness* tertingi. Tiga buah skenario uji coba yang akan dilakukan, antara lain:

4.5.1 Perancangan Pengujian Berdasarkan Jumlah Populasi

Uji coba dilakukan berdasarkan banyaknya *popsize* yang bertujuan untuk mengetahui pengaruh *popsize* terhadap nilai *fitness*. Ukuran populasi yang diujikan adalah angka kelipatan 10 mulai dari 10 sampai 100. Masing-masing uji coba dilakukan sebanyak 10 kali. Tabel 4.13 menunjukkan rancangan uji coba populasi.

Tabel 4.13 Rancangan uji coba banyaknya *popsize.*

Popsize	Nilai *fitness*										Rata-rata nilai *fitness*
	Pengujian ke-i										
	1	2	3	4	5	6	7	8	9	10	
10											
20											
30											
40											
50											
60											
70											
80											
90											
100											

4.5.2 Perancangan Pengujian Berdasarkan Kombinasi Nilai *Crossover Rate* dan *Mutation Rate*

Uji coba dilakukan berdasarkan nilai Cr dan Mr yang bertujuan untuk mengetahui pengaruh nilai Cr dan Mr terhadap nilai *fitness*. Nilai Cr dan Mr yang digunakan adalah bilangan antara 0 sampai 1. Masing-masing uji coba dilakukan sebanyak 10 kali. Tabel 4.14 menunjukkan uji coba Cr dan Mr.

Tabel 4.14 Rancangan uji coba Cr dan Mr.

Cr; Mr	Nilai *fitness*										Rata-rata nilai *fitness*
	Pengujian ke-i										
	1	2	3	4	5	6	7	8	9	10	
0.1 ; 0.9											
0.2 ; 0.8											
0.3 ; 0.7											
0.4 ; 0.6											
0.5 ; 0.5											
0.6 ; 0.4											
0.7 ; 0.3											
0.8 ; 0.2											
0.9 ; 0.1											

4.5.3 Perancangan Pengujian Berdasarkan Jumlah Generasi

Uji coba dilakukan berdasarkan jumlah generasi yang bertujuan untuk mengetahui pengaruh jumlah generasi terhadap nilai *fitness*. Jumlah generasi yang digunakan adalah kelipatan 10 mulai dari 10 sampai 100. Masing-masing uji coba dilakukan sebanyak 10 kali. Tabel 4.15 menunjukkan uji coba jumlah generasi.

Tabel 4.15 Rancangan uji coba jumlah generasi.

Generasi	Nilai *fitness*										Rata-rata nilai *fitness*
	Pengujian ke-i										
	1	2	3	4	5	6	7	8	9	10	
10	0.123456	0.123456	0.123456	0.123456	0.123456	0.123456	0.123456	0.123456	0.123456	0.123456	0.123456
20											
30											
40											
50											
60											
70											
80											
90											
100											

BAB 5 IMPLEMENTASI

Pada bab implementasi menjelaskan tentang hasil dari pembuatan program berdasarkan perancangan sistem pada bab sebelumnya dan telah menerapkan konsep algoritme genetika di dalam penelitian ini.

5.1 Struktur Class

Penerapan program yang dilakukan pada penelitian ini menggunakan bahasa pemrograman Java. Pemrograman dilakukan dengan menggunakan *NetBeans* sebagai aplikasi pembantu selama pembuatan program. Adapun susunan *class* yang dibuat terdiri dari:

a. Algoritme genetika merupakan *class* yang di dalamnya terdapat method dan kode

dari penerapan algoritme genetika, mulai dari pembentukkan populasi (*popsize*) proses reproduksi, evaluasi hingga seleksi.

b. Data Lokasi merupakan *class* yang digunakan untuk menyimpan data jarak antar lokasi penumpang dan waktu tempuh ke dalam bentuk array, dimana nantinya akan digunakan untuk menghitung total jarak dan total waktu dalam proses perhitungan nilai *fitness*.

c. MTSP *Travel* merupakan *class* yang digunakan untuk menjalankan program dimana di dalamnya terdapat inisialisasi dari nilai Cr, Mr dan *popsize*.

5.2 Potongan *Source Code*

Dari hasil implementasi program yang telah dilakukan, maka pada sub bab ini akan dijelaskan mengenai potongan *source code* yang telah dibuat berdasarkan perancangan algoritme, antarmuka dan perancangan pengujian yang dijelaskan pada bab 4 analisis dan perancangan.

5.2.1 *Source Code* Data Jarak dan Waktu

Data jarak dan waktu yang digunakan dalam penelitian ini di inputkan ke dalam class Data Lokasi, kemudian ditampilkan ke dalam program. Source Code untuk data jarak dan waktu ditunjukkan pada Source Code 5.1.

```
public class DataLokasi {
        double[][] jarakLokasi, waktuTempuh;
        String[] lokasi;

DataLokasi() {
        this.jarakLokasi = new double[][]{
        {0.0, 8.9, 4.9, 5.3, 4.2, 6.9, 3.0, 3.1, 1.8, 5.2},
        {8.9, 0.0, 13.7, 11.7, 11.9, 11.0, 11.7, 12.3, 8.9, 13.1},
        {4.9, 13.7, 0.0, 5.2, 3.6, 6.7, 3.4, 4.0, 5.2, 4.1},
         {5.3, 11.7, 5.2, 0.0, 1.5, 1.8, 3.3, 3.4, 6.9, 2.1},
         {4.2, 11.9, 3.6, 1.5, 0.0, 4.0, 1.6, 2.3, 5.8, 1.0},
         {6.9, 11.0, 6.7, 1.8, 4.0, 0.0, 5.4, 5.2, 8.4, 4.3},
         {3.0, 11.7, 3.4, 3.3, 1.6, 5.4, 0.0, 0.7, 3.5, 2.1},
         {3.1, 12.3, 4.0, 3.4, 2.3, 5.2, 0.7, 0.0, 4.6, 3.5},
         {1.8, 8.9, 5.2, 6.9, 5.8, 8.4, 3.5, 4.6, 0.0, 5.7},
         {5.2, 13.1, 4.1, 2.1, 1.0, 4.3, 2.1, 3.5, 5.7, 0.0}};
```

```
this.waktuTempuh = new double[][]{
{0.0, 0.57, 0.28, 0.27, 0.30, 0.40, 0.27, 0.23, 0.10, 0.37},
{0.57, 0.0, 0.93, 0.72, 0.75, 0.67, 0.68, 0.70, 0.42, 0.78},
{0.28, 0.93, 0.0, 0.28, 0.18, 0.35, 0.20, 0.23, 0.40, 0.22},
{0.27, 0.72, 0.28, 0.0, 0.32, 0.12, 0.17, 0.17, 0.40, 0.13},
{0.30, 0.75, 0.18, 0.32, 0.0, 0.18, 0.10, 0.13, 0.38, 0.20},
{0.40, 0.67, 0.35, 0.12, 0.18, 0.0, 0.23, 0.23, 0.35, 0.20},
{0.27, 0.68, 0.20, 0.17, 0.10, 0.23, 0.0, 0.15, 0.23, 0.10},
{0.23, 0.70, 0.23, 0.17, 0.13, 0.23, 0.15, 0.0, 0.22, 0.17},
{0.10, 0.42, 0.40, 0.40, 0.38, 0.35, 0.23, 0.22, 0.0, 0.28},
{0.37, 0.78, 0.22, 0.13, 0.20, 0.20, 0.10, 0.17, 0.28, 0.0}};

 String[] abc = new String[]{
         "Jl. Pisang Kipas No.10",
         "Perum. Banjararum Asri Blok B no.21",
         "Perum. Bukit Cemara  Tidar Blok B no.33",
         "Jl. Bareng Tengah V no.681",
         "Jl. Klampok Kasri 2 no.231",
         "Jl. Pasar Besar no.58",
         "Jl. Bendungan Sutami no.2B",
         "Jl.Veteran no.2",
         "Jl Candi Panggung Barat",
         "Jl. Halimun no.8, Dieng"};
         this.lokasi = abc;
            }
}
```

Source Code 5.2.1 Data Jarak dan Waktu

5.2.2 Source Code Pembentukan Populasi (*Popsize*)

Proses pembentukan populasi *(popsize*) dilakukan setelah memasukkan nilai-nilai parameter. Proses ini dilakukan dengan membentuk jumlah individu sebanyak masukkan populasi (*popsize*) yang diinginkan serta membentuk urutan gen menjadi

kromosom secara acak (*random)*. *Source code* pembentukkan populasi awal ditunjukkan pada *Source Code* 5.2

```
        void inisialisasi() {


        System.out.println("Betul");
        int[] inisialisasi = new int[panjangKromosom];
        for (int j = 0; j < panjangKromosom; j++) {
        inisialisasi[j] = j;
            }
        //tukar" indeks
Random random = ThreadLocalRandom.current();
System.arraycopy(inisialisasi,0,parent[0],0, panjangKromosom);
 for (int i = 1; i < popsize; i++) {
System.arraycopy(inisialisasi,0,parent[i],0, panjangKromosom);
 for (int j = panjangKromosom - 1; 0 < j; j--) {
 int indeksTukar = random.nextInt(j + 1);
 int sementara = parent[i][j];
        parent[i][j] = parent[i][indeksTukar];
        parent[i][indeksTukar] = sementara;
        }
        }
        System.out.println("Populasi Awal");
        for (int i = 0; i < popsize; i++) {
        System.out.print("P" + (i + 1) + " ");
        for (int j = 0; j < panjangKromosom; j++) {
        System.out.print((parent[i][j] + 1) + " ");
              }
        System.out.println("");
            }
  }
```

Source Code 5. 2 Pembentukan Populasi *(Popsize)*

5.2.3 Source Code Proses Crossover

Pada penelitian ini proses *crossover* menggunakan teknik *one-cut* point yang dilakukan dengan cara menukarkan urutan gen dari titik tertentu antara dua individu yang dipilih secara *random*. Banyaknya individu baru yang dihasilkan pada proses *crossover* ditentukan dari besaran *crossover rate* yang telah dimasukkan sebelumnya. Proses *one-cut point crossover* ditunjukkan pada *Source Code* 5.3.

```
        void crossover() {
System.out.println("Banyak child crossover :" + banyakchildCrossover);
        int counter = 0; int counterTampil = 0;
        while (counter < banyakchildCrossover) {

        Random random = ThreadLocalRandom.current();
        int cutPoint = random.nextInt(panjangKromosom - 2) + 1;
        int indeksParent1 = random.nextInt(panjangKromosom - 1);
        int indeksParent2 = random.nextInt(panjangKromosom - 1);

        while (indeksParent1 == indeksParent2) {
        indeksParent2 = random.nextInt(panjangKromosom - 1);
        }

        int[] offSpring1 = new int[panjangKromosom];
        int[] offSpring2 = new int[panjangKromosom];

        for (int x = 0; x < cutPoint; x++) {
        offSpring1[x] = parent[indeksParent1][x];
            offSpring2[x] = parent[indeksParent2][x];
        }

        //cek sudah ada atau belum
        for (int x = cutPoint; x < panjangKromosom; x++) {
           for (int p = 0; p < panjangKromosom; p++) {
              //jika nilai dari parent 2 belum ada pada offSpring 1, masukkan
              if (!cekSudahAda(offSpring1, parent[indeksParent2][p])) {
                 offSpring1[x] = parent[indeksParent2][p];
              }
```

```
                    if (!cekSudahAda(offSpring2, parent[indeksParent1][p])) {
                        offSpring2[x] = parent[indeksParent1][p];
                    }
                }
            }

            child[counter] = offSpring1;
            child[counter + 1] = offSpring2;
            counter = counter + 2;



            System.out.println("Parent yang dipilih : " + (indeksParent1 + 1) + " dan " +
(indeksParent2 + 1) + ". Cutpoint : " + cutPoint);
            System.out.print("P" + (indeksParent1 + 1) + " ");
            for (int j = 0; j < panjangKromosom; j++) {
                System.out.print((parent[indeksParent1][j] + 1) + " ");
            }
            System.out.println("");
            System.out.print("P" + (indeksParent2 + 1) + " ");
            for (int j = 0; j < panjangKromosom; j++) {
                System.out.print((parent[indeksParent2][j] + 1) + " ");
            }
            System.out.println("");

            for (int j = 0; j < panjangKromosom; j++) {
                System.out.print((offSpring1[j] + 1) + " ");
            }
            System.out.println("");
            counterTampil++;

            if (counterTampil != banyakchildCrossover) {
                for (int j = 0; j < panjangKromosom; j++) {
                    System.out.print((offSpring2[j] + 1) + " ");
                }
```

```
                System.out.println("");
                counterTampil++;
            }
        }
    }
```

Source Code 5. 3 Proses *Crossover*

5.2.4 *Source Code* Proses *Mutation*

Proses *mutation* merupakan proses reproduksi selanjutnya di mana pada penelitian ini digunakan teknik *reciprocal exchange*. Teknik *reciprocal exchange* dilakukan dengan memilih dua titik gen secara random dari satu individu acak yang nantinya titik dari gen tersebut akan ditukarkan posisinya. Jumlah individu baru yang dihasilkan disesuaikan dengan besaran *mutation rate* yang telah dimasukkan sebelumnya. Proses *mutation* ditunjukkan pada *Source Code* 5.4.

```
void mutasi() {
    int counter = banyakchildCrossover;

    while (counter < banyakchildCrossover + banyakchildMutation) {
        Random random = ThreadLocalRandom.current();
        int parentTukar = random.nextInt(popsize - 1);
        int indeksTukar1 = random.nextInt(panjangKromosom - 1);
        int indeksTukar2 = random.nextInt(panjangKromosom - 1);
        while (indeksTukar1 == indeksTukar2) {
            indeksTukar2 = random.nextInt(panjangKromosom - 1);
        }

        int[] offSpring = new int[panjangKromosom];
        System.arraycopy(parent[parentTukar], 0, offSpring, 0, panjangKromosom);

        int sementara = offSpring[indeksTukar2];
        offSpring[indeksTukar2] = offSpring[indeksTukar1];
        offSpring[indeksTukar1] = sementara;

        child[counter] = offSpring;
```

```
                counter++;


                System.out.println("Parent yang dipilih : P" + parentTukar + ". ");
                System.out.println("Indeks yang ditukar : " + (indeksTukar1 + 1) + ", " +
(indeksTukar2 + 1));
                for (int j = 0; j < panjangKromosom; j++) {
                   System.out.print((parent[parentTukar][j] + 1) + " ");
                }
                System.out.println("");
                for (int j = 0; j < panjangKromosom; j++) {
                   System.out.print((offSpring[j] + 1) + " ");
                }
                System.out.println("");
             }
             System.out.println("");
  }
```

Source Code 5. 4 Proses Mutasi

5.2.5 Source Code Proses Evaluasi dan Seleksi

Proses evaluasi merupakan sebuah tahapan dimana seluruh individu yang telah dihasilkan baik dari pembentukkan populasi maupun hasil dari reproduksi akan dicari nilai totalJaraknya totalWaktu menentukkan nilai *fitness* dari masing-masing individu tersebut. Dilanjutkan proses seleksi yang merupakan tahap terakhir dari satu kali siklus proses algoritme genetika dengan menentukan individu terbaik berdasarkan nilai *fitness*-nya. Teknik yang digunakan dalam proses seleksi adalah seleksi *elitism* yang menentukkan individu terbaik berdasarkan nilai *fitness* terbesar ke terkecil sebanyak populasi sebelumnya. Proses evaluasi dan seleksi ditunjukkan pada *Source Code* 5.5

```
        double[] evaluasi(int[][] kromosom) {
            DecimalFormat df = new DecimalFormat("#.#########");
            double[] fitnessParentChild = new double[popsize + child.length];
            double[] totalJarak = new double[kromosom.length];
            double[] totalWaktu = new double[kromosom.length];
```

```
            //System.out.print("Jarak : ");
            for (int i = 0; i < popsize + child.length; i++) {
              double jarak = 0, waktu = 0;
              for (int j = 0; j < panjangKromosom - 1; j++) {
                jarak = jarak + dataLokasi.jarakLokasi[kromosom[i][j]][kromosom[i][j + 1]];
                waktu = waktu + dataLokasi.waktuTempuh[kromosom[i][j]][kromosom[i][j
+ 1]];


                //System.out.print(dataLokasi.jarakLokasi[kromosom[i][j]][kromosom[i][j
+ 1]] + "+ ");
              }


              totalJarak[i] = jarak +
dataLokasi.jarakLokasi[kromosom[i][panjangKromosom - 1]][kromosom[i][0]];
              totalWaktu[i] = waktu +
dataLokasi.waktuTempuh[kromosom[i][panjangKromosom - 1]][kromosom[i][0]];


              //System.out.println("Total Jarak : " + totalJarak[i]);
              fitnessParentChild[i] = 100 / totalJarak[i] + totalWaktu[i];
            }


            System.out.println("Evalusi");
            System.out.println("      Kromosom \t \t    Total Jarak\t   Total Waktu
\t\tFitness");
            for (int i = 0; i < kromosom.length; i++) {
              if (i < popsize) {
                System.out.print(" P" + (i + 1) + " ");
              } else {
                System.out.print(" C" + (i + 1) + " ");
              }
              for (int j = 0; j < panjangKromosom; j++) {
                System.out.print((kromosom[i][j] + 1) + " ");
              }
              System.out.print("\t " + df.format(totalJarak[i]) + "\t \t" +
df.format(totalWaktu[i]) + "\t \t");
              System.out.println(df.format(fitnessParentChild[i]) + " ");
```

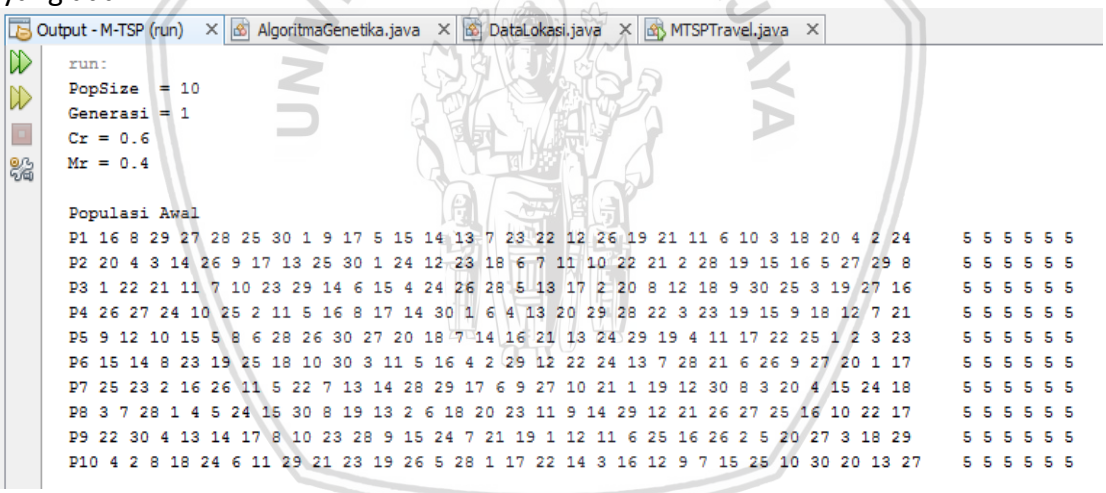
```
    }

    System.out.println("");
    return fitnessParentChild;
}

void seleksi(int[][] parentdanChild, double[] fitnessParentdanChild) {
    int[] indeksUrut = urutkan(fitnessParentdanChild);

    //tampilkan hasil pengurutan
    System.out.println("Diurutkan berdasarkan fitness");
    for (int i = 0; i < popsize; i++) {
        for (int j = 0; j < panjangKromosom; j++) {
            System.out.print(parentdanChild[indeksUrut[i]][j] + " ");
        }
        System.out.println("\t" + fitnessParentdanChild[indeksUrut[i]]);
    }

    //ambil sebanyak popsize, jadikan parent generasi selanjutnya
    for (int p = 0; p < popsize; p++) {
        parent[p] = parentdanChild[indeksUrut[p]];
    }
}

boolean cekSudahAda(int[] array, int angka) {
    boolean ada = false;
    for (int i = 0; i < array.length; i++) {
        if (angka == array[i]) {
            ada = true;
            break;
        }
    }
    return ada;
}
```

```
        //method untuk mengurutkan. input berupa array fitness, outputnya indeks fitness yang sudah urut
        int[] urutkan(double[] fitnessParentdanChild) {
            int[] indeksUrut = new int[fitnessParentdanChild.length];
            for (int i = 0; i < fitnessParentdanChild.length; i++) {
                indeksUrut[i] = i;
            }

            System.out.println("Hasil Seleksi :  ");
            double[] fitnessUrut = new double[fitnessParentdanChild.length];
            System.arraycopy(fitnessParentdanChild, 0, fitnessUrut, 0, fitnessParentdanChild.length);

        //      System.out.println("");
        //      System.out.println("Sebelum diurutkan : ");
        //      for(int i=0; i<fitnessUrut.length; i++){
        //          System.out.println(fitnessUrut[i]);
        //      }
            for (int x = fitnessUrut.length; x >= 0; x--) {
                for (int y = 0; y < x - 1; y++) {
                    if (fitnessUrut[y] < fitnessUrut[y + 1]) {
                        double sementara = fitnessUrut[y];
                        int indeksSementara = indeksUrut[y];
                        fitnessUrut[y] = fitnessUrut[y + 1];
                        indeksUrut[y] = indeksUrut[y + 1];
                        fitnessUrut[y + 1] = sementara;
                        indeksUrut[y + 1] = indeksSementara;
                    }
                }
            }

            fitnessTertinggi = fitnessUrut[0];
```

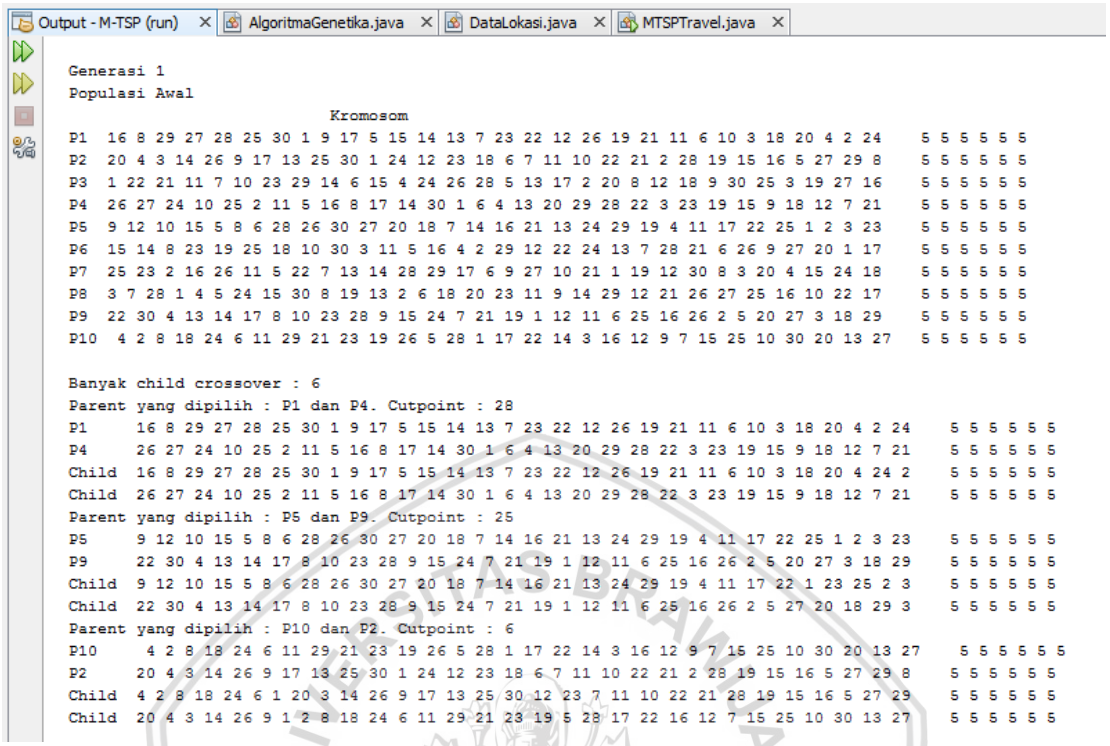
```
        //      System.out.println("");
        //      System.out.println("Setelah diurutkan");
        //      for(int i=0; i<fitnessUrut.length; i++){
        //          System.out.println(fitnessParentdanChild[i]);
        //      }
            return indeksUrut;
          }
}
```

5.3 Implementasi Antarmuka Pengguna

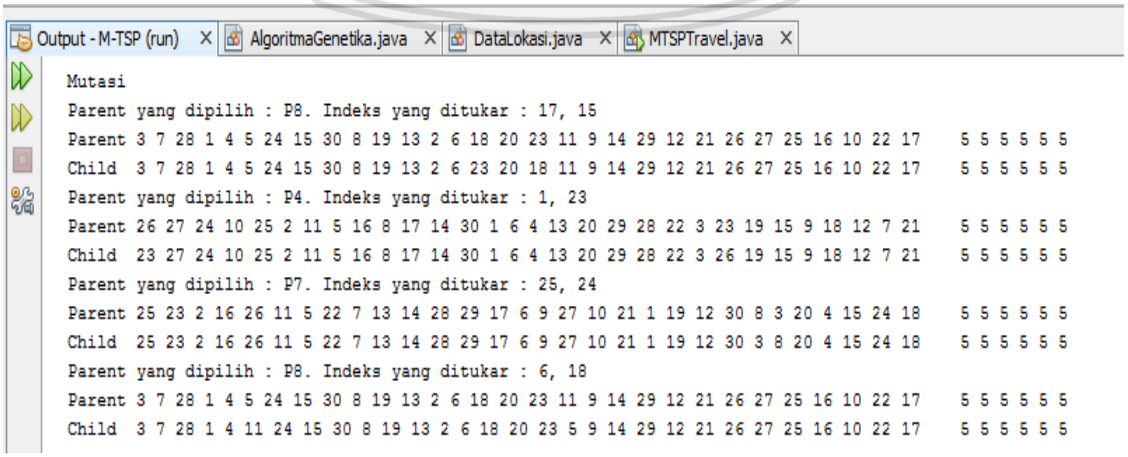
Hasil implementasi pada sistem penjemputan penumpang *travel* menggunakan algoritme genetika menampilkan jumlah parameter algoritme genetika yang akan digunakan dalam proses perhitungan berupa *popsize*, generasi, nilai cr dan mr, serta menampilkan proses reproduksi, evaluasi dan seleksi dari masing-masing individu yang ada.



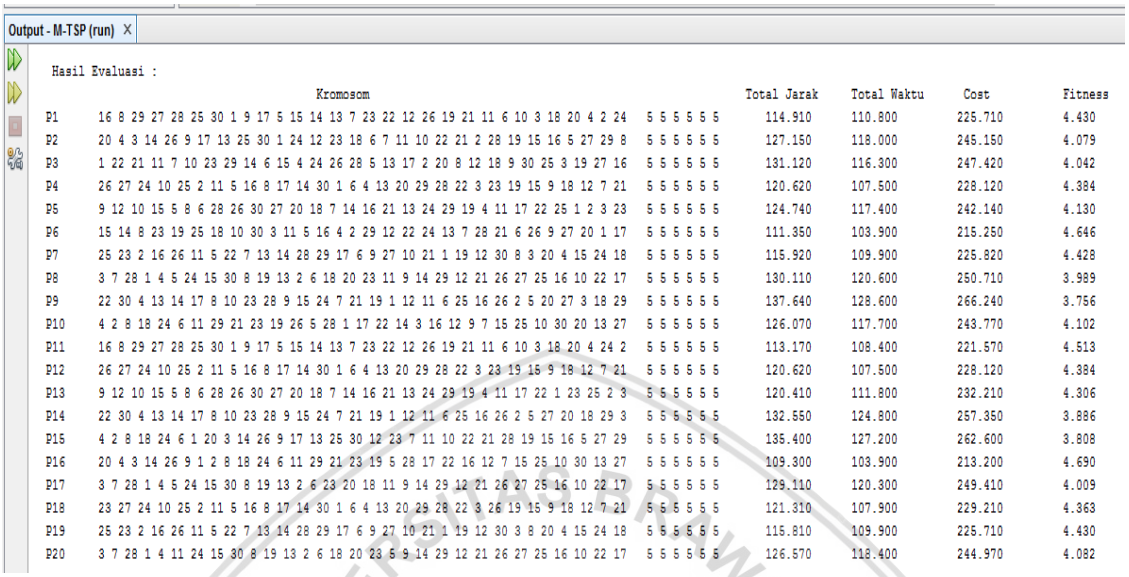
Gambar 5.3 Antarmuka Populasi Awal



Gambar 5.4 Antarmuka Proses Crossover



Gambar 5.4 Antarmuka Proses Mutasi

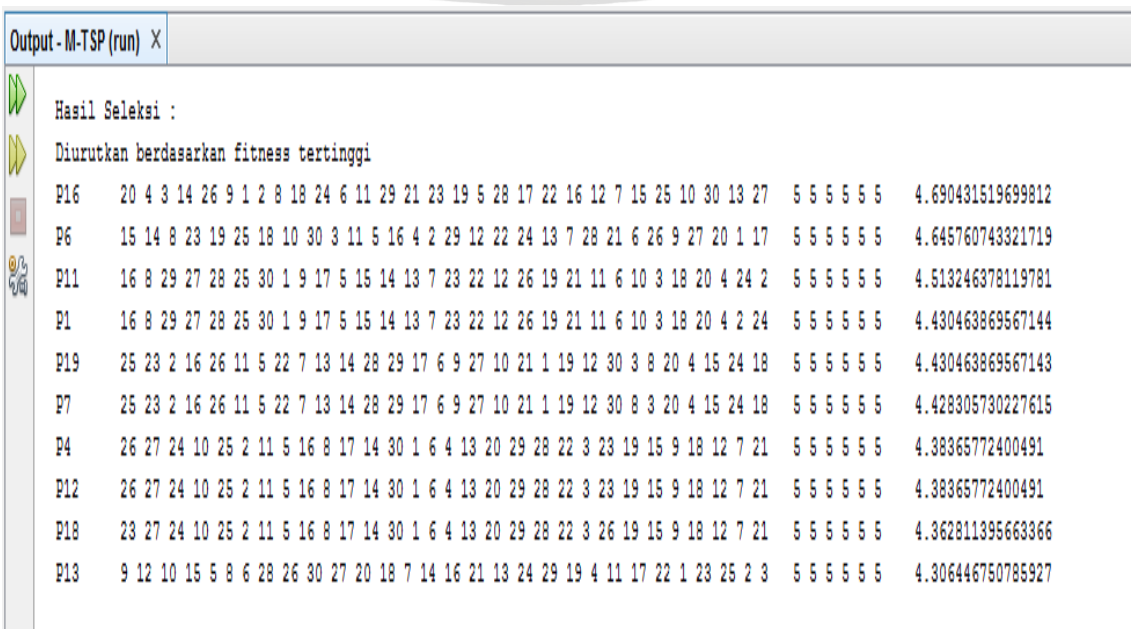


Output - M-TSP (run)

Hasil Evaluasi :

	Kromosom	Total Jarak	Total Waktu	Cost	Fitness
P1	16 8 29 27 28 25 30 1 9 17 5 15 14 13 7 23 22 12 26 19 21 11 6 10 3 18 20 4 2 24 5 5 5 5 5 5	114.910	110.800	225.710	4.430
P2	20 4 3 14 26 9 17 13 25 30 1 24 12 23 18 6 7 11 10 22 21 2 28 19 15 16 5 27 29 8 5 5 5 5 5 5	127.150	118.000	245.150	4.079
P3	1 22 21 11 7 10 23 29 14 6 15 4 24 26 28 5 13 17 2 20 8 12 18 9 30 25 3 19 27 16 5 5 5 5 5 5	131.120	116.300	247.420	4.042
P4	26 27 24 10 25 2 11 5 16 8 17 14 30 1 6 4 13 20 29 28 22 3 23 19 15 9 18 12 7 21 5 5 5 5 5 5	120.620	107.500	228.120	4.384
P5	9 12 10 15 5 8 6 28 26 30 27 20 18 7 14 16 21 13 24 29 19 4 11 17 22 25 1 2 3 23 5 5 5 5 5 5	124.740	117.400	242.140	4.130
P6	15 14 8 23 19 25 18 10 30 3 11 5 16 4 2 29 12 22 24 13 7 28 21 6 26 9 27 20 1 17 5 5 5 5 5 5	111.350	103.900	215.250	4.646
P7	25 23 2 16 26 11 5 22 7 13 14 28 29 17 6 9 27 10 21 1 19 12 30 8 3 20 4 15 24 18 5 5 5 5 5 5	115.920	109.900	225.820	4.428
P8	3 7 28 1 4 5 24 15 30 8 19 13 2 6 18 20 23 11 9 14 29 12 21 26 27 25 16 10 22 17 5 5 5 5 5 5	130.110	120.600	250.710	3.989
P9	22 30 4 13 14 17 8 10 23 28 9 15 24 7 21 19 1 12 11 6 25 16 26 2 5 20 27 3 18 29 5 5 5 5 5 5	137.640	128.600	266.240	3.756
P10	4 2 8 18 24 6 11 29 21 23 19 26 5 28 1 17 22 14 3 16 12 9 7 15 25 10 30 20 13 27 5 5 5 5 5 5	126.070	117.700	243.770	4.102
P11	16 8 29 27 28 25 30 1 9 17 5 15 14 13 7 23 22 12 26 19 21 11 6 10 3 18 20 4 24 2 5 5 5 5 5 5	113.170	108.400	221.570	4.513
P12	26 27 24 10 25 2 11 5 16 8 17 14 30 1 6 4 13 20 29 28 22 3 23 19 15 9 18 12 7 21 5 5 5 5 5 5	120.620	107.500	228.120	4.384
P13	9 12 10 15 5 8 6 28 26 30 27 20 18 7 14 16 21 13 24 29 19 4 11 17 22 1 23 25 2 3 5 5 5 5 5 5	120.410	111.800	232.210	4.306
P14	22 30 4 13 14 17 8 10 23 28 9 15 24 7 21 19 1 12 11 6 25 16 26 2 5 27 20 18 29 3 5 5 5 5 5 5	132.550	124.800	257.350	3.886
P15	4 2 8 18 24 6 1 20 3 14 26 9 17 13 25 30 12 23 7 11 10 22 21 28 19 15 16 5 27 29 5 5 5 5 5 5	135.400	127.200	262.600	3.808
P16	20 4 3 14 26 9 1 2 8 18 24 6 11 29 21 23 19 5 28 17 22 16 12 7 15 25 10 30 13 27 5 5 5 5 5 5	109.300	103.900	213.200	4.690
P17	3 7 28 1 4 5 24 15 30 8 19 13 2 6 23 20 18 11 9 14 29 12 21 26 27 25 16 10 22 17 5 5 5 5 5 5	129.110	120.300	249.410	4.009
P18	23 27 24 10 25 2 11 5 16 8 17 14 30 1 6 4 13 20 29 28 22 3 26 19 15 9 18 12 7 21 5 5 5 5 5 5	121.310	107.900	229.210	4.363
P19	25 23 2 16 26 11 5 22 7 13 14 28 29 17 6 9 27 10 21 1 19 12 30 3 8 20 4 15 24 18 5 5 5 5 5 5	115.810	109.900	225.710	4.430
P20	3 7 28 1 4 11 24 15 30 8 19 13 2 6 18 20 23 5 9 14 29 12 21 26 27 25 16 10 22 17 5 5 5 5 5 5	126.570	118.400	244.970	4.082

Gambar 5.5 Antarmuka Hasil Evaluasi



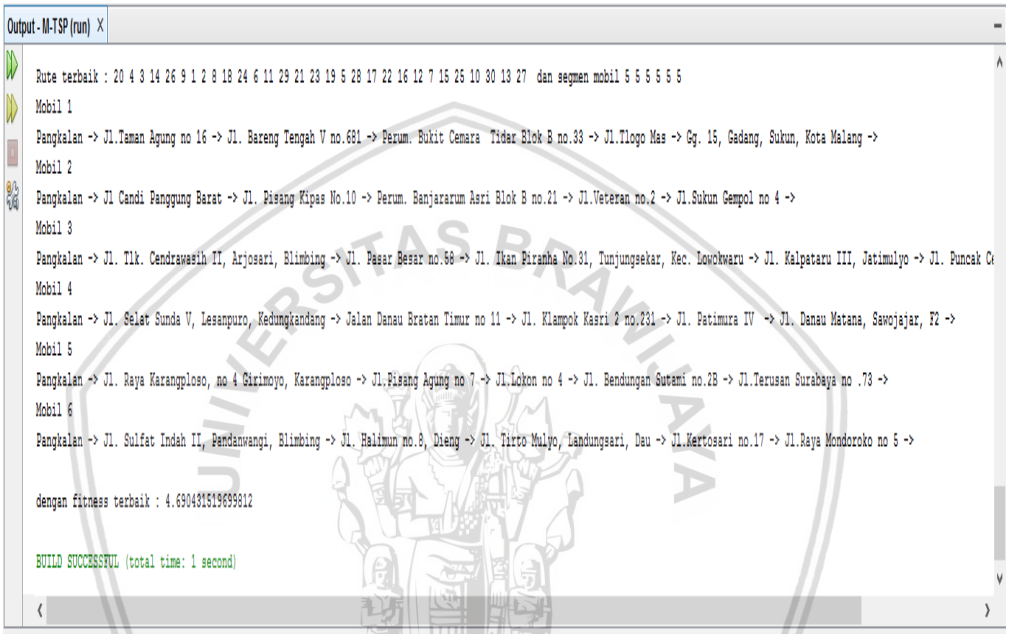
Output - M-TSP (run)

Hasil Seleksi :

Diurutkan berdasarkan fitness tertinggi

P16	20 4 3 14 26 9 1 2 8 18 24 6 11 29 21 23 19 5 28 17 22 16 12 7 15 25 10 30 13 27 5 5 5 5 5 5	4.690431519699812
P6	15 14 8 23 19 25 18 10 30 3 11 5 16 4 2 29 12 22 24 13 7 28 21 6 26 9 27 20 1 17 5 5 5 5 5 5	4.645760743321719
P11	16 8 29 27 28 25 30 1 9 17 5 15 14 13 7 23 22 12 26 19 21 11 6 10 3 18 20 4 24 2 5 5 5 5 5 5	4.5132463781191781
P1	16 8 29 27 28 25 30 1 9 17 5 15 14 13 7 23 22 12 26 19 21 11 6 10 3 18 20 4 2 24 5 5 5 5 5 5	4.430463869567144
P19	25 23 2 16 26 11 5 22 7 13 14 28 29 17 6 9 27 10 21 1 19 12 30 3 8 20 4 15 24 18 5 5 5 5 5 5	4.430463869567143
P7	25 23 2 16 26 11 5 22 7 13 14 28 29 17 6 9 27 10 21 1 19 12 30 8 3 20 4 15 24 18 5 5 5 5 5 5	4.428305730227615
P4	26 27 24 10 25 2 11 5 16 8 17 14 30 1 6 4 13 20 29 28 22 3 23 19 15 9 18 12 7 21 5 5 5 5 5 5	4.38365772400491
P12	26 27 24 10 25 2 11 5 16 8 17 14 30 1 6 4 13 20 29 28 22 3 23 19 15 9 18 12 7 21 5 5 5 5 5 5	4.38365772400491
P18	23 27 24 10 25 2 11 5 16 8 17 14 30 1 6 4 13 20 29 28 22 3 26 19 15 9 18 12 7 21 5 5 5 5 5 5	4.362811395663366
P13	9 12 10 15 5 8 6 28 26 30 27 20 18 7 14 16 21 13 24 29 19 4 11 17 22 1 23 25 2 3 5 5 5 5 5 5	4.306446750785927

Gambar 5.6 Antarmuka Hasil Seleksi



m
uka Hasil Solusi Rute Terbaik

BAB 6 PENGUJIAN DAN PEMBAHASAN

Pada bab pengujian dan pembahasan akan dibahas mengenai pengujian yang telah dilakukan pada sistem optimasi penentuan rute optimal penjemputan penumpang *travel* berdasarkan perancangan pengujian yang telah dijelaskan sebelumnya.

6.1 Mekanisme Pengujian

Mekanisme pengujian terhadap sistem penentuan rute optimal penjemputan penumpang *travel* dibagi menjadi 3 jenis pengujian parameter. Pengujian parameter ini terdiri dari pengujian jumlah populasi (*popsize*), pengujian jumlah generasi, dan pengujian kombinasi *crossover rate* (*cr*) dan *mutation rate* (*mr*). Tiap skenario

dilakukan pengujian sebanyak 10 kali. Setelah dilakukan pengujian akan dicari nilai rata-rata *fitness* dan *fitness* tertinggi dari masing-masing populasi. Dari nilai parameter terbaik inilah yang nantinya akan digunakan untuk mencari hasil penentuan rute optimal penjemputan penumpang *travel*.

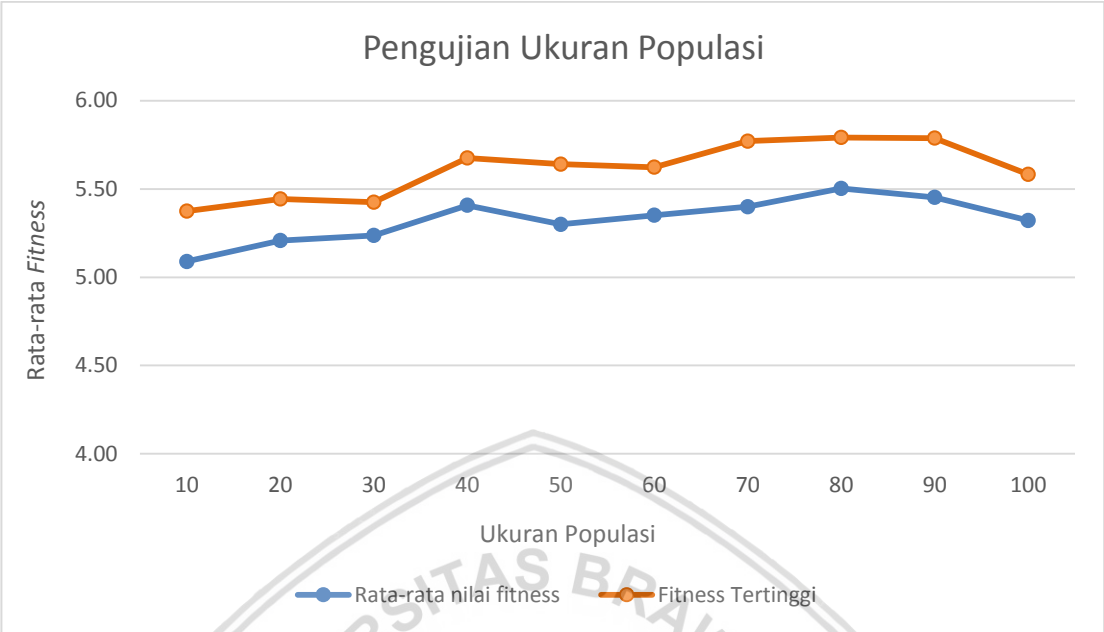
6.2 Pengujian Jumlah Populasi

Dalam tahap ini jenis pengujian yang pertama dilakukan adalah pengujian jumlah populasi yang bertujuan untuk mengetahui pengaruh ukuran populasi (*popsize*) terhadap nilai *fitness*. Perancangan uji coba pada bab sebelumnya menjelaskan bahwa, didapatkan hasil uji coba ukuran populasi sebanyak 10 kali percobaan dengan menggunakan ukuran populasi kelipatan 10 hingga 100. Setiap ukuran populasi dilakukan sebanyak 10 kali proses percobaan. Pada pengujian ini menggunakan parameter dengan banyak generasi 10, dengan kombinasi *cr* dan *mr* sebesar 0.6 dan 0.4. Hasil pengujian ukuran populasi ditunjukkan pada Tabel 6.1.

Tabel 6.1 Hasil Pengujian Ukuran Populasi

Popsize	Nilai *fitness* Pengujian ke-i					*Fitness* Tertinggi	Rata-rata nilai *fitness*
	1	2	3	…	10		
10	4.93974	4.98679	5.23478	…	5.27065	5.37490	5.08916
20	5.08053	5.38590	5.42064	…	5.19022	5.44366	5.20829
30	5.33903	5.27176	5.42535	…	5.12374	5.42535	5.23720
40	5.35217	5.29633	5.57134	…	5.22876	5.67601	5.40821
50	5.30983	5.32652	5.18269	…	5.46956	5.64079	5.29956
60	5.28961	5.42446	5.29914	…	5.39811	5.62395	5.35172
70	5.40424	5.77101	5.28667	…	5.15490	5.77101	5.39941
80	5.53006	5.53526	5.52730	…	5.79249	5.79249	5.50331
90	5.37808	5.55124	5.51785	…	5.07315	5.78871	5.35295
100	5.10955	5.34359	5.47915	…	5.17866	5.58316	5.28172

Pada Tabel 6.1 terdapat nilai rata-rata *fitness* paling optimum yang terletak pada ukuran populasi 80 dengan rata-rata nilai *fitness* sebesar 5.50331, sedangkan rata-rata nilai *fitness* terkecil terletak pada ukuran populasi 10 dengan rata-rata nilai *fitness*nya 5.089159. Selain itu, terdapat pula ukuran populasi *fitness* tertinggi terdapat pada ukuran populasi 80 dengan dengan rata-rata nilai *fitness* sebesar 5.78871. Keseluruhan nilai rata-rata *fitness* dan *fitness* tertinggi melalui 10 kali pengujian dapat dilihat pada Gambar 6.1.



Gambar 6.1 Grafik Hasil Pengujian Ukuran Populasi

Kesimpulan dari pengujian ini adalah rata-rata nilai *fitness* yang dihasilkan dipengaruhi oleh besarnya ukuran populasi. Dilihat pada perkembangan nilai *fitness*nya, dari ukuran populasi 10 hingga 100 grafik perlahan semakin naik. Namun terjadi penurunan rata-rata nilai *fitness* pada ukuran populasi 50, 90, dan 100. Ukuran populasi yang terbaik berada pada ukuran populasi 80 yang menjadikannya sebagai titik optimum.

6.3 Pengujian Jumlah Generasi

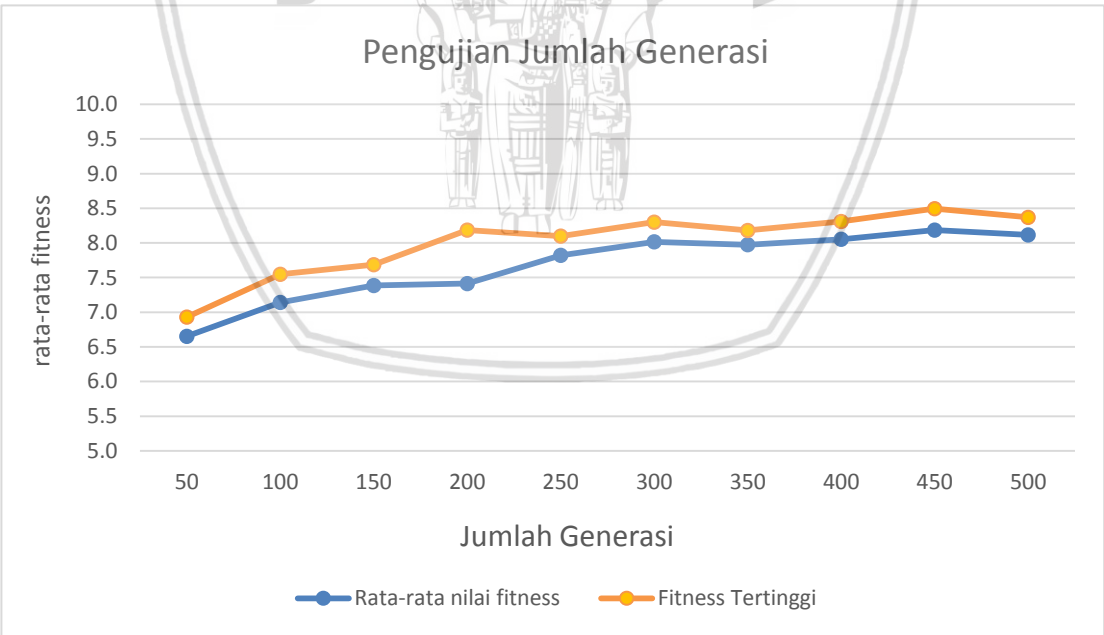
Tahap pengujian selanjutnya yaitu pengujian jumlah generasi yang digunakan untuk mengetahui pengaruh jumlah generasi terhadap nilai *fitness*. Pengujian ini dilakukan dengan memasukkan jumlah generasi mulai dari 50 sampai dengan 500 sesuai dengan kelipatan 50. Kemudian memasukkan parameter lainnya yaitu *popsize* sebesar 80 sesuai dengan ukuran populasi terbaik pada pengujian populasi sebelumnya dan kombinasi *cr* dan *mr* sebesar 0.6 dan 0.4. Hasil pengujian jumlah generasi ditunjukkan pada Tabel 6.2.

Tabel 6. 2 Hasil Pengujian Jumlah Generasi

Generasi	Nilai *fitness*					*Fitness* Tertinggi	Rata-rata nilai *fitness*
	Pengujian ke-i						
	1	2	3	…	10		
50	6.30756	6.43169	6.57635	…	6.70556	6.92953	6.65177

100	6.69478	7.24061	7.14286	...	7.23118	7.54704	7.14124
150	7.09132	7.38498	7.25163	...	7.52798	7.68550	7.38518
200	7.15147	7.07376	7.22055	...	7.27983	8.18472	7.41261
250	7.44269	7.27749	7.97333	...	8.38792	8.38792	7.84946
300	7.84991	8.14910	7.89359	...	7.93529	8.51937	8.03778
350	7.84912	7.94218	8.09119	...	7.98516	8.27914	7.98408
400	7.81013	7.93626	8.07914	...	7.96431	8.30894	8.05271
450	8.18583	8.23995	8.49370	...	8.29572	8.50937	8.18613
500	8.05819	8.11426	7.97885	...	7.81849	8.17206	7.94258

Pada Tabel 6.2 terdapat nilai *fitness* paling optimum yang terletak pada jumlah generasi 450 dengan rata-rata nilai *fitness* sebesar 8.18457. Sedangkan jumlah generasi dengan rata-rata nilai *fitness* terkecil terletak pada ukuran populasi 50 dengan rata-rata nilai *fitness*nya sebesar 6.65177. Selain itu, terdapat pula *fitness* tertinggi terdapat pada jumlah generasi 300 dengan rata-rata nilai *fitness* sebesar 8.51937. Keseluruhan nilai rata-rata *fitness* dan *fitness* tertinggi melalui 10 kali pengujian dapat dilihat pada Gambar 6.2.



Gambar 6.2 Grafik Hasil Pengujian Jumlah Generasi

Kesimpulan dari pengujian ini adalah rata-rata nilai *fitness* yang dihasilkan dipengaruhi oleh jumlah generasi. Dilihat pada perkembangan nilai *fitness*nya, dari jumlah generasi 50 hingga 500 grafik cenderung mengalami peningkatan sepanjang

proses pengujian. Namun terjadi penurunan rata-rata nilai *fitness* pada jumlah generasi 350 dan 500. Terjadinya peningkatan dan kemudian mengalami penurunan nilai *fitness* karena sifat algoritme genetika dimana kromosom diperoleh secara acak sehingga memungkinkan nilai *fitness* yang didapatkan pada kromosom yang lebih kecil. Pada kasus penjemputan *travel* dengan menggunakan jumlah penumpang sebanyak 30 dan jumlah percobaan sebanyak 10 kali, algoritme genetika menghasilkan solusi terbaik pada jumlah generasi 450.

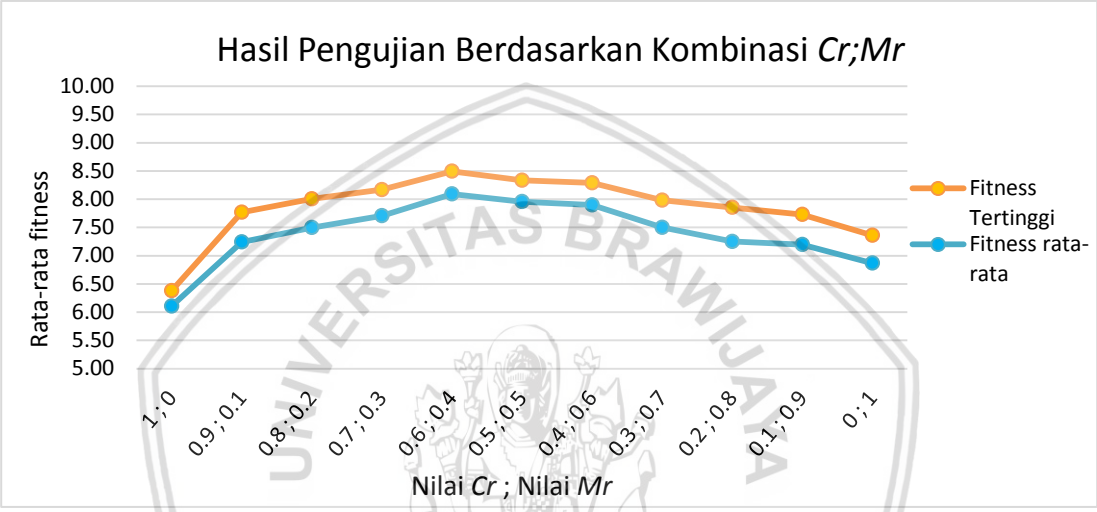
6.4 Pengujian Kombinasi *Crossover Rate* (*cr*) dan *Mutation Rate* (*mr*)

Pengujian berikutnya yang dilakukan adalah pengujian nilai kombinasi *crossover rate* (*cr*) dan *mutation rate* (*mr*) yang bertujuan untuk mengukur dan mengetahui pengaruh nilai *Cr* dan *Mr* terhadap nilai *fitness*. Masing-masing uji coba dilakukan sebanyak 10 kali dengan kombinasi nilai *cr* dan *mr* mulai dari 0 hingga 1. Nilai parameter yang digunakan adalah nilai terbaik pada pengujian sebelumnya yaitu *popsize* sebesar 80, generasi sebesar 450. Hasil pengujian kombinasi *crossover rate* dan *mutation rate* ditunjukkan pada Tabel 6.3.

Tabel 6. 3 Tabel Hasil Pengujian Kombinasi *Crossover Raite* dan *Mutation Rate*

Cr ; Mr	Nilai *fitness*					*Fitness* Tertinggi	Rata-rata nilai *fitness*
	Pengujian ke-i						
	1	2	3	…	10		
1 ; 0	6.04157	5.98874	5.97836	…	5.92260	6.38040	6.10694
0.9 ; 0.1	7.17566	7.76940	7.57972	…	7.02260	7.76940	7.24362
0.8 ; 0.2	7.31154	8.00617	7.21207	…	7.30776	8.00617	7.49899
0.7 ; 0.3	7.27908	7.25624	7.47518	…	8.07201	8.16876	7.70759
0.6 ; 0.4	7.56452	7.23903	7.92870	…	8.37983	8.49642	8.09338
0.5 ; 0.5	7.15623	7.28579	7.45756	…	8.29967	8.33505	7.94406
0.4 ; 0.6	7.41866	7.84898	7.75225	…	8.02784	8.28920	7.89580
0.3 ; 0.7	7.81311	7.12454	7.57117	…	7.98234	7.98234	7.50073
0.2 ; 0.8	6.96098	7.15408	7.11221	…	7.35919	7.85602	7.25086
0.1 ; 0.9	7.49457	7.20346	7.41340	…	6.95319	7.73036	7.19724

Pada Tabel 6.3 terdapat nilai parameter paling optimum yang terletak pada nilai *crossover rate* 0.6 dan *mutation rate* 0.4 dengan rata-rata nilai *fitness* yang dihasilkan yaitu 8.09338, sedangkan rata-rata nilai *fitness* terkecil terletak pada nilai *crossover rate* 1 dan *mutation rate* 0 dengan dengan rata-rata nilai *fitness*nya 6.10694. Selain itu, terdapat pula nilai *fitness* tertinggi terdapat pada nilai *crossover rate* 0.6 dan *mutation rate* 0.4 dengan nilai *fitness* 8.09338. Keseluruhan nilai rata-rata *fitness* dan *fitness* tertinggi melalui 10 kali pengujian ditunjukkan pada gambar 6.3.



Gambar 6.3 Grafik Hasil Pengujian Kombinasi Nilai *cr* dan *mr*

Kesimpulan dari pengujian ini adalah rata-rata nilai *fitness* yang dihasilkan dipengaruhi oleh kombinasi nilai *cr* dan *mr*. Dilihat dari grafik, kombinasi nilai *cr* dan *mr* menunjukkan bahwa rata-rata *fitness* terendah cenderung berada pada kombinasi nilai cr dan mr yang tidak seimbang, sedangkan rata-rata *fitness* tertinggi cenderung berada pada kombinasi nilai *cr* dan *mr* yang memiliki nilai seimbang. Hal ini dikarenakan dalam algoritme genetika penentuan *crossover rate* dan *mutation rate* dapat digunakan untuk menentukan sebuah keseimbangan dalam eksplorasi dan eksplotasi. Uji coba dengan menggunakan nilai *cr* dan *mr* yang jika dijumlah bernilai 1, bertujuan agar populasi memiliki individu yang stabil setelah dilakukannya proses reproduksi. Pada kasus M-TSP penjemputan *travel* dengan menggunakan jumlah penumpang sebanyak 30 dan jumlah percobaan sebanyak 10 kali, algoritme genetika menghasilkan solusi terbaik pada kombinasi *crossover rate* 0.6 dan *mutation rate* 0.4.

6.5 Pengujian Menggunakan Data Uji

Pengujian berikutnya yang dilakukan adalah pengujian menggunakan data uji yaitu data baru yang berbeda dari data sebelumnya. Data yang digunakan adalah data alamat penumpang sebanyak 10 dan dilakukan sebanyak 10 kali uji coba. Nilai parameter algoritma genetika yang digunakan adalah nilai terbaik berdasarkan pengujian sebelumnya yaitu ukuran *popsize* sebesar 80, jumlah generasi sebesar 450, kombinasi *crossover rate* 0.6 dan *mutation rate* 0.4. Hasil pengujian menggunakan data uji ditunjukkan pada Tabel 6.4.

Tabel 6. 4 Tabel Hasil Pengujian Menggunakan Data Uji

Pop size	Generasi	*Cr* ; *Mr*	Nilai *fitness*					*Fitness* Tertinggi	Rata-rata nilai *fitness*
			Pengujian ke-i						
			1	2	3	…	10		
80	450	0.6 ; 0.4	22.64493	22.98851	21.76752	…	22.98851	22.98851	22.83205

6.6 Pengujian Jumlah Salesman

Pengujian berikutnya yang dilakukan adalah pengujian jumlah salesman. Pengujian pertama menggunakan 2 salesman dengan jumlah penumpang sebanyak 10. Pengujian ini menggunakan parameter algoritme genetika berupa nilai terbaik berdasarkan pengujian sebelumnya yaitu ukuran *popsize* sebesar 80, jumlah generasi sebesar 450, kombinasi *crossover rate* 0.6 dan *mutation rate* 0.4. Hasil pengujian menggunakan 2 salesman ditunjukkan pada Tabel 6.4.

Tabel 6. 4 Tabel Hasil Pengujian Menggunakan 2 sales

Segmen mobil	Mobil 1	Mobil 2	*Fitness*
5-5	Pangkalan -> Jl. Tlogomas No.5 -> Jl. Tata Surya No.2, Dinoyo -> Jl.Bendungan Wonogiri No.4 Sumbersari -> Jl.Sidoarjo no.10 Gading Kasri, Klojen -> Jl.Buring no.18 Oro-oro Dowo.	Pangkalan -> Jl.Nusa Indah No.10 Lowokwaru -> Jl.Lembang No.106 Samaan, Klojen -> Jl.Cakalang No.146 Blimbing -> Jl.Puri Palma no 2 Blimbing -> Jl.Wisnuwardhana No.20 A. Mangliawan, Pakis.	25.100

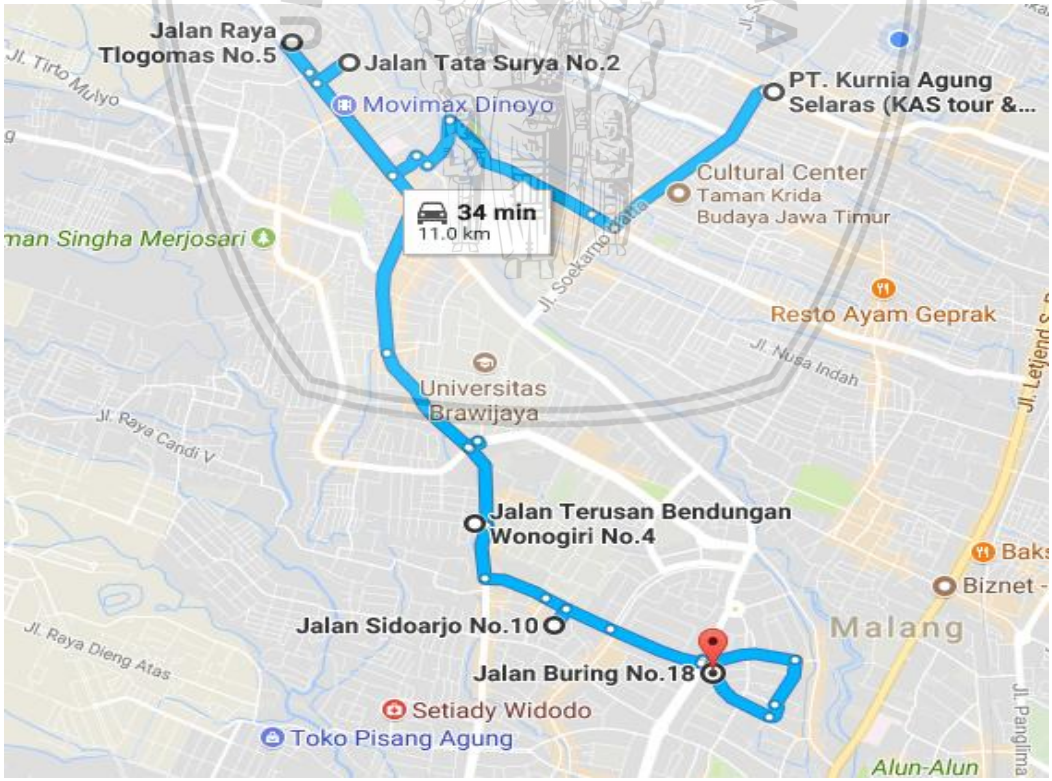
Berdasarkan tabel 6.4 didapatkan hasil *fitness* sebesar 22.214, dimana masing-masing mobil menjemput 5 orang penumpang. Dalam permasalahan ini, menggunakan 10 alamat penumpang dan kapasitas untuk 1 mobil hanya dapat menampung 5 penumpang saja, sehingga segmen mobil untuk 2 sales adalah 5-5 dengan jumlah penumpang sebanyak 10 orang. Pengujian selanjutnya yaitu pengujian menggunakan 3 salesman dengan jumlah penumpang sebanyak 10. Hasil pengujian menggunakan 3 salesman ditunjukkan pada Tabel 6.5.

Tabel 6. 5 Tabel Hasil Pengujian Menggunakan 3 sales

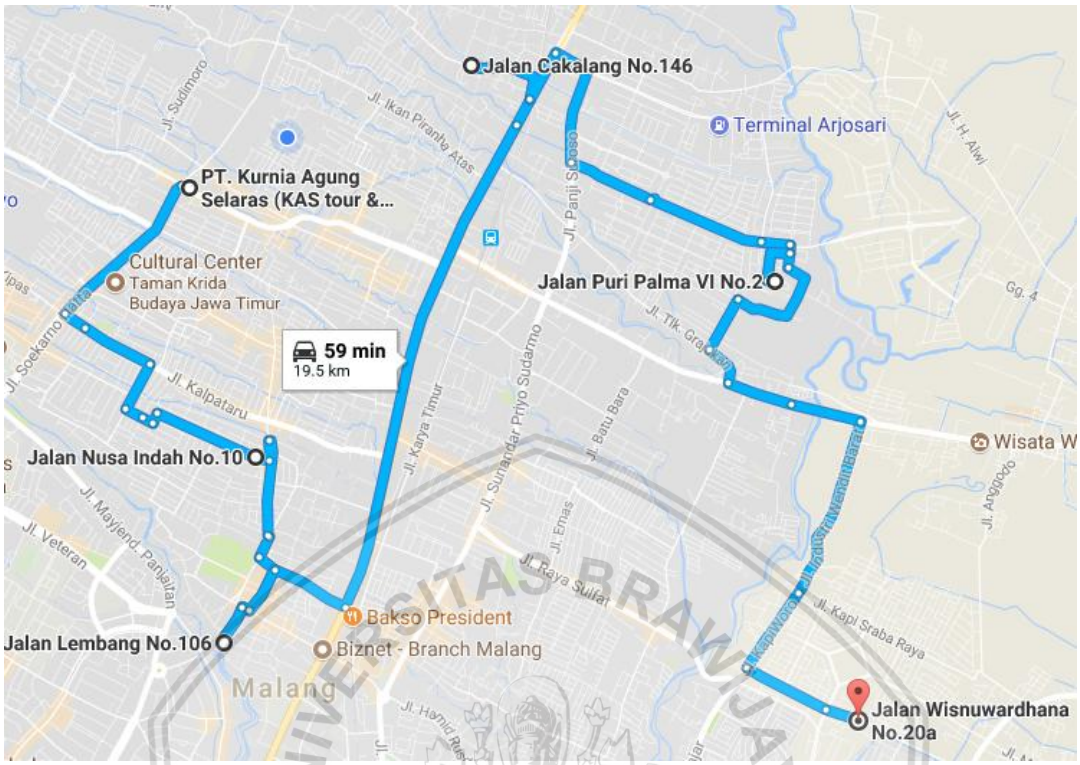
Segmen mobil	Mobil 1	Mobil 2	Mobil 3	*Fitness*
3-3-4	Pangkalan -> Jl. Tata Surya No.2, Dinoyo -> Jl.Buring no.18 Oro-oro Dowo -> Jl.Wisnuwardhana No.20 A. Mangliawan, Pakis.	Pangkalan -> Jl. Tlogomas No.5 -> Jl.Cakalang No.146 Blimbing -> Jl.Puri Palma no 2 Blimbing.	Pangkalan -> Jl.Nusa Indah No.10 Lowokwaru -> Jl.Lembang No.106 Samaan, Klojen -> Jl.Sidoarjo no.10 Gading Kasri, Klojen -> Jl.Bendungan Wonogiri No.4 Sumbersari.	22.187
3-4-3	Pangkalan -> Jl. Tlogomas No.5 -> Jl.Cakalang No.146 Blimbing -> Jl.Puri Palma no.2 Blimbing.	Pangkalan -> Jl.Bendungan Wonogiri No.4 Sumbersari -> Jl.Sidoarjo no.10 Gading Kasri, Klojen -> Jl.Buring no.18 Oro-oro Dowo -> Jl.Wisnuwardhana No.20 A. Mangliawan, Pakis	Pangkalan -> Jl.Nusa Indah No.10 Lowokwaru -> Jl.Lembang No.106 Samaan, Klojen -> Jl. Tata Surya No.2, Dinoyo .	22.411

4-3-3	Pangkalan -> Jl.Cakalang No.146 Blimbing -> Jl.Nusa Indah No.10 Lowokwaru -> Jl.Wisnuwardhana No.20 A. Mangliawan, Pakis -> Jl.Puri Palma no 2 Blimbing.	Pangkalan -> Jl.Lembang No.106 Samaan, Klojen -> Jl. Tlogomas No.5 -> Jl. Tata Surya No.2, Dinoyo.	Pangkalan -> Jl.Sidoarjo no.10 Gading Kasri, Klojen -> Jl.Bendungan Wonogiri No.4 Sumbersari -> Jl.Buring no.18 Oro-oro Dowo.	23.883

Berdasarkan tabel 6.5 didapatkan hasil *fitness* terbaik sebesar 23.883 dengan segmen mobil 4-3-3. Dilihat dari hasil kedua tabel dapat diketahui bahwa dengan menggunakan 2 sales untuk 10 alamat penumpang mendapatkan hasil solusi yang lebih optimal. Hal ini dikarenakan dengan penggunaan jumlah mobil yang lebih sedikit maka semakin sedikit pula biaya tempuh yang diperlukan. Berikut merupakan rute terbaik penjemputan penumpang *travel* menggunakan 2 sales. Rute dari mobil 1 ditunjukan pada gambar 6.4 dan rute mobil 2 ditunjukkan pada gambar 6.5.



Gambar 6.4 Rute Mobil 1



Gambar 6.4 Rute Mobil 2

BAB 7 KESIMPULAN DAN SARAN

7.1 Kesimpulan

Dari hasil pengujian yang telah dilakukan, maka didapatkan kesimpulan sebagai berikut :

1. Pada penelitian ini, implementasi metode algoritme genetika telah berhasil diterapkan terhadap optimasi M-TSP pada penentuan rute optimal penjemputan penumpang *travel* di mana diperoleh solusi akhir berupa rute terbaik dalam penjemputan penumpang *travel*. Adapun proses reproduksi yang digunakan yaitu *one cut point crossover* dan *reciprocal exchange mutation*. Pada proses seleksi menggunakan metode *elitism selection*
2. Representasi kromosom yang digunakan yaitu representasi permutasi dikarenakan representasi permutasi dinilai mampu menyelesaikan dengan efisien dan paling sesuai untuk diterapkan dalam permasalahan M-TSP penentuan rute optimal penjemputan penumpang *travel*. Setiap kromosom dalam populasi memiliki dua segmen, untuk segmen pertama panjang kromosom menunjukkan banyaknya lokasi penjemputan penumpang *travel* dan segmen kedua panjang kromosom berdasarkan jumlah sales yang melakukan penjemputan.

3. Penyelesaian masalah untuk optimasi MTSP pada penentuan rute optimal penjemputan penumpang *travel* dipengaruhi oleh beberapa parameter algoritme genetika yang ada di dalam hasil pengujian. Parameter algoritme genetika digunakan yaitu ukuran *popsize*, nilai *crossover rate (cr), mutation rate (mr)* dan jumlah generasi.

7.2 Saran

Berdasarkan penelitian yang telah dilakukan dengan beberapa sub pembahasan disertai dengan hasil analisis pengujian sistem, didapatkan saran untuk pengembangan lebih lanjut yaitu sebagai berikut :

1. Penambahan proses pengujian dalam parameter algoritme genetika agar didapatkan parameter yang lebih baik dan mendapatkan solusi yang lebih optimal.
2. Pada penelitian selanjutnya diharapkan dapat mengoptimasi proses penjemputan penumpang dengan mempertimbangkan kondisi jalan dan kemacetan jalan raya yang dilewati sehingga waktu penjemputan penumpang menjadi lebih optimal.
3. Pada kromosom segmen kedua yaitu jumlah penjemputan penumpang setiap sales sebaiknya menggunakan jumlah nilai yang dinamis, sehingga pembagian penjemputan setiap sales juga lebih optimal.

DAFTAR PUSAKA

Alves, Raulcezar M.F. & Lopez, Carlos R., 2015. *Using Genetic Algorithms to minimize the distance and balance the routes for the Multiple Traveling Salesman Problem.* IEEE Congress on Evolutionary Computation (CEC). DOI: 10.1109/CEC.2015.7257285

Fatkhiyah, E., 2012**.** *Rancangan Proses Training Untuk Mendukung Penentuan Kualitas Air Minum Kemasan*. Prosiding Seminar Nasional Aplikasi Sains & Teknologi (SNAST). Yogyakarta, 3 November 2012.

Gajera,V.,et al, 2016.*An Effective Multi Objective Task Scheduling Algorithm using Min-Max Normalization in Cloud Computing.* 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT).

Hasibuan, M.D.A.C & Lusiana, 2015. *Pencarian Rute Terbaik pada Travelling Salesman Problem (TSP) menggunakan Algoritme Genetika pada Dinas Kebersihan dan Pertamanan kota Pekanbaru*. SATIN - Sains dan Teknologi Informasi, Vol. 1, No. 1, Juni 2015.

Haupt, R. & Haupt, E. S., 2004. *Practical Genetic Algorithms*. USA: John Wiley & Sons.

Nurhumam, S. D. & Mahmudy, W. F., 2008. *Optimasi Multi Travelling Salesman Problem (M-TSP) pada Mobil Patroli Polisi dengan Algoritme Heuristic Assigment Fisher-Jaikumar dan Algoritme A*.* Kursor, Volume 4, pp. 15-22.

Mahmudy, W.F., 2008. *Optimasi Multi Travelling Salesman Problem (M-TSP) Menggunakan Algoritma Genetika*. Seminar Nasional Basic Science V.

Mahmudy, W.F., 2013. *Algoritme Evolusi. Program Teknologi Informasi dan Ilmu Komputer*. Universitas Brawijaya, Malang.

Mahmudy, Wayan Firdaus (2015). *Modul Algoritme Evolusi.* Program Teknologi Informasi dan Ilmu Komputer (PTIIK) Universitas Brawijaya.

Mayuliana, N.K, Kencana, E.N & Harini, L.P.I (2017). *Penyelesaian Multi Traveling Salesman Problem Dengan Algoritma Genetika*. E-Jurnal Matematika Vol. 6 (1) - pp. 1-6. DOI: https://doi.org/10.24843/MTK.2017.v06.i01.p141.

Nisaa, Khoiron (2013). *Optimasi Rute Angkutan kota Malang dengan Penerapan Algoritme Genetika.* Universitas Brawijaya.

Ponraj, R & Amalanathan, G (2014). *Optimizing Multiple Travelling Salesman Problem Considering The Road Capacity*. Journal of Computer Science 10 (4): 680-688, 2014
ISSN: 1549-3636

Putri, F.B., Mahmudy,W.F., & Ratnawati,D.E. (2015). *Penerapan Algoritma Genetika Untuk Vehicle Routing Problem with Time Window (VRPTW) Pada Kasus Optimasi Distribusi Beras Bersubsidi*. Jurnal Skripsi.DORO: Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol. 5, no. 1.

Samudra,L & Muklash,I (2013). *Penentuan Rute Optimal Pada Kegiatan Penjemputan Penumpang Travel Menggunakan Ant Colony System*. Jurnal Sains Dan Seni Pomits Vol. 2, No.1, (2013) 1-6.

Saptaningtyas, FY (2015). *Multi Traveling Salesman Problem (M-TSP) Dengan Algoritme Genetika Untuk Menentukan Rute Loper Koran Di Agen Surat Kabar.* Jurusan Pendidikan Matematika FMIPA UNY

Sari,RN & Mahmudy,WF (2015). *Penyelesaian Multiple Travelling Salesperson Problem (M-TSP) Dengan Algoritme Genetika : Studi Kasus Pendistribusian Air Mineral*. Jurnal Skripsi.DORO: Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol. 5, no. 14. Malang.

Singh,S & Lodhi,E.A (2014). *Comparison Study of Multiple Traveling Salesmen Problem using Genetic Algorithm*. IJCSNS International Journal of Computer Science and Network Security, VOL.14 No.7, July 2014.

Sulistiyorini, R & Mahmudy, WF (2015). *Penerapan algoritme genetika untuk permasalahan optimasi distribusi barang dua tahap*. DORO: Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol. 5, no. 12.

Wang,Y, Chen,Y & Lin,Y (2016*). Memetic algorithm based on sequential variable neighborhood descent for the minmax multiple traveling salesman problem.* Computers & Industrial Engineering (2016), doi: http://dx.doi.org/10.1016/j.cie.2016.12.017.

Widodo, A. W. & Mahmudy, W. F., (2010). *Penerapan Algoritme Genetika Pada Sistem Rekomendasi Wisata Kuliner*. Jurnal Ilmiah KURSOR, 5(0216-0544), pp. 205-211.

Wijayaningrum, Vn & Mahmudy, W.F (2016). *Optimization of Ship's Route Scheduling Using Genetic Algorithm*. Indonesian Journal of Electrical Engineering and Computer Science, vol. 2, no. 1, pp. 180-186.

Yuli, F.(2015). *Multi Traveling Salesman Problem (M-TSP) Dengan Algoritme Genetika Untuk Menentukan Rute Loper Koran Di Agen Surat Kabar.*